Published in Proceedings of the International Conference on Internet of Things and Big Data, 391-398, 2016, Rome, Italy; http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0005953803910398

Security Incident Information Exchange for Cloud Services

Christian Frøystad[1], Erlend Andreas Gjære[1], Inger Anne Tøndel[1] and Martin Gilje Jaatun[1]
[1]*SINTEF ICT, Strindvegen 4, Trondheim, Norway*
{*Christian.Froystad, ErlendAndreas.Gjare, IngerAnne.Tondel, Martin.G.Jaatun*}*@sintef.no*

Keywords: incident response, cloud computing, accountability

Abstract: The complex provider landscape in cloud computing makes incident handling difficult, as Cloud Service Providers (CSPs) with end-user customers do not necessarily get sufficient information about incidents that occur at upstream CSPs. In this paper, we argue the need for commonly agreed-upon incident information exchanges between providers as a means to improve accountability of CSPs. The discussion considers several technical challenges and non-technical aspects related to improving the situation for incident response in cloud computing scenarios. In addition, we propose a technical implementation which can embed standard representation formats for incidents in notification messages, built over a publish-subscribe architecture, and a web-based dashboard for handling the incident workflow.

1 INTRODUCTION

Cloud computing offers its users a significant amount of benefits such as increased agility, reliability, easier and better scalability and elasticity, maintenance, device and location independence, and reduced cost (Kalloniatis et al., 2014). Due to this, the popularity of cloud infrastructure is understandably on the rise among both smaller companies and multinational enterprises alike, further enabling start-up companies to innovate with rapidly growing customer bases without costly IT investments up-front.

While the benefits of cloud computing are well known, there are still drawbacks or challenges which need to be taken into account by stakeholders looking into adoption of such infrastructure. This paper focusses on *incident response*, which is the process of handling the occurrence of an incident from detection, through analysis, containment, eradication & recovery and preparation (Grobauer and Schreck, 2010). These activities have increased in complexity since the time when servers were physical machines running a single system for one organisation, possibly at their own physical premises, too. A set of security issues related to incident handling *in the cloud* were examined by Grobauer & Schreck (Grobauer and Schreck, 2010) back in 2010, who then called for more research in several areas. In the years which have passed since this paper, there has only been published a little amount of research addressing those challenges. Most of this, however, is mainly concerned with digital forensics in the cloud, or more traditional incident response scenarios, and nothing on the perspective of dealing with complex cloud provisioning chains. In general, we find a lack of academic attention paid to incident management between independent companies and organizations involved in cloud computing, and to giving the involved parties sufficient access to related data and event sources in a timely manner.

Sharing of incident information has typically been based on one-to-one trust relationships between Computer Security Incident Response Team (CSIRT) members in the relevant organisations. The actual exchange of incident information normally happen by means of email, phone, incident trackers, help desk systems, conference calls and face to face. With the advent of cloud computing, however, the human element is much less prominent. A cloud service can be made up of a chain of providers where none of the CSIRT members have ever communicated directly with a representative from any of the other providers. In addition, an incident in the cloud may need to involve different parts of the provider chain, potentially even in an automated, real-time fashion, to minimise business disruption (Gjære et al., 2014). This sets new requirements to the way incident response needs to be managed and supported by tools which are able to communicate effectively across rapidly changing constellations of organisations.

An essential challenge is that there is no single part of the supply chain which has access to all events and all areas to monitor, thus nobody can immediately see the full picture. In a survey conducted by Torres (Torres, 2014), *little visibility* into system/endpoint configurations/vulnerabilities as such was considered one of the top hindrances to effective incident response in their organisation. The cloud actors therefore need to be able to communicate efficiently in order to provide each other with information to ease detection or assist responding to an incident. It has also been claimed that attackers are better at handling information sharing than those protecting services and systems (Horne, 2014). This adds to the importance of providing good tools and solutions to incident handlers.

Laws are powerful incentives for changing behaviors in entire industries within a country. When large unions, like the United States or the European Union, introduces quite similar laws, this affects the entire western world and also, to some degree, the rest of the world (Greenleaf, 2012). The introduction of new regulations in Europe, like the General Data Protection Regulation (GDPR) and Network Information Security (NIS), increases the need for an effective way of exchanging incident information. Due to the substantial fines mandated by the GDPR, service providers are given an incentive to ensure accurate and timely notification about breaches relating to personal information. Laws are not only an incentive, but sometimes also a hindrance or at least an obstacle. The difference between laws covering personal information, could complicate information exchange.

In this paper we provide an overview of business aspects (section 2) and technical challenges (section 3), which directly impact incident response abilities for cloud computing scenarios. Moreover, in section 4, we propose a notification message format which has room for both standards-based and customised contents, and demonstrate it in use through a prototype of our *IncidentTracker* system. The prototype features a scalable architecture of *IncidentTracker* instances, each implementing a common web interface specification for communication between providers, as well as a simple web-based dashboard for managing the workflow related to notification messages. An overall discussion of the approach is provided in section 6, while section 7 concludes the paper.

2 BUSINESS ASPECTS

In order for an incident exchange solution to be useful, it needs to be adopted by businesses and CSPs. Given how most businesses strive to improve their financial results, it is likely that for the system to be adopted, one of the following criteria must be fulfilled:

- Use of the solution results in reduced costs or increased revenue – directly or indirectly
- Actors are required by law to use a system similar to this solution

More efficient incident response, e.g. by increased automation of response to common and simple incidents (Metzger et al., 2011), can result in a financial benefit. Additionally, professional incident notification schemes can strengthen the provider's reputation as an accountable and trustworthy provider. However, the sharing of incident information can also be considered risky for a provider. Traditionally, incident information has mostly been shared directly between people with a direct trust relationship, i.e., who know each other personally. In vendor and outsourcing relationships, organizations thus take many measures to build trust in order to facilitate incident information exchange (Jaatun and Tøndel, 2015). It may not be possible to achieve the same level of trust between organizations as between individuals. Cloud providers, however, cannot rely on individual trust relationships, but need to establish trust on an organizational level and exchange incident information based on that trust.

Bandyophyay et al. (Bandyopadhyay et al., 2009) divides costs of cyber-incidents into two broad categories: primary and secondary losses, where primary losses refer to direct loss and operating loss and secondary losses refer to any second-degree effects that is indirectly triggered by information concerning the security of the company (e.g. reputation damage or credit rating). Several factors may impact the cost. As an example, Bandyophyay et al. (Bandyopadhyay et al., 2009) explains how a decision to report the incident to an insurance company can increase the cost associated with the incident: the reporting may result in the breach becoming known to external parties, and thus secondary losses is experienced. Cloud providers may be reluctant towards sharing incident information for similar reasons. In addition, there may be a need to share information on the incident before the incident is fully understood, including the cause of the incident. Revealing incident information may thus, in the worst case, give information to any attackers on the impact of their attack. Technological solutions that provide secure means to distribute incident informa-

tion can to some extent reduce the risk of sharing incident information, as providers can have an improved overview of who has received what information regarding the incident. Terms regarding sharing and receiving incident information can also be covered in contracts.

3 FUNCTIONAL CONSIDERATIONS

As a basis for our proposed specification, we have investigated functional aspects of how incident information should be represented and shared. The following provides an overview of how these aspects relate to cloud computing scenarios in particular.

3.1 One Format to Rule Them All?

In order to ensure that every part of the cloud supply chain is able to understand the information sent, we believe it is necessary to agree on a definition of an incident and its data elements. This would allow for easier development and deployment of new systems as well as increase interoperability, as one would not have to conform to multiple different definitions and data sets, but can rely on one base format. At the same time, if an organization needs a different representation internally, this should still be possible as long as it is feasible to translate this representation into the common representation.

Schneier (Schneier, 2014) claims that "*Incidents aren't standardized; they're all different*." NIST (Cichonski et al., 2012) further state that each CSIRT team must choose their own list of required data elements, based on factors like team model, team structure and how the team defines an incident. This raises the question of whether it is actually possible to represent every incident in one format, or if it is a better approach to allow multiple formats and thus allow for specialization. By using only one format that is able to represent everything, one could, theoretically, know that all conforming implementations would be able to understand any incident received and be able to handle it automatically, if desired. On the other hand, the format would be very complex, as it needs to include mechanisms to represent all possible relevant information for any incident imaginable.

We have found the Incident Object Description Exchange Format (IODEF) (Danyliw et al., 2007) to be a comprehensive format for expressing incidents. While IODEF in practice can support any property included in other relevant formats, including Structured Threat Information eXpression (STIX) (Barnum, 2012), eCrime (Cain and Jevans, 2010), and Cyber Observable eXpression (CybOX) (The MITRE Corporation, 2015), it also brings along significant challenges. The format is complex, making it difficult to understand for humans, and the amount of extensions that would need to be created in order for the format to be fully usable for automatic handling of incidents, is large.

On the other hand, an option can be to only transfer unstructured text messages between parties in the cloud supply chain. This would reduce the solution to be a secure "email" system, and thus in accordance with how incident information is mostly exchanged today (Frøystad, 2014). The ability to provide free text would allow the parties to exchange any information required, but they would have to agree on a special formatting for the text if planning to support automatic handling of incidents. This would make it easier to implement the solution, and thus reduce initial adoption costs. Notable drawbacks of this approach include fragmentation in message formatting, leading to potential difficulties for humans in interpreting or encoding incident information.

A middle ground would be to have a small base format, with the ability to represent the most common information in a simple way and providing a structured way of attaching other incident formats such as those mentioned above. This allows for reuse of existing incident formats, specialization, incremental implementation, and flexibility to support newer formats.

3.2 Notification

At some point, the CSP experiencing an incident needs to notify its cloud customers – that is other providers in the cloud supply chain buying services from the CSP. This could be because required by law, legal agreements with the cloud customer or because the CSP strives to be accountable. One approach is to decide which notifications to send *a-priori* by the mutual agreement between provider and customer, which each make sure that they exchange the necessary information to fulfil the relevant laws and provide sufficient data for information exchange to be useful.

Another approach could be that the provider decides which notifications to send, without the cloud customer's influence. This could include notifications with a core message like: "We have been breached, but your data was not compromised."

A hybrid, combining the two, would allow the mutual agreement to be created by the provider listing which types of incidents the cloud customer is allowed to access, and the cloud customer subscribing

to the incident types he needs. In order to allow the cloud provider to fulfil any legal obligations to notify the cloud customer, he could, when obligated by law, send such notifications whether the cloud customer subscribes to them or not.

Figure 1 shows an example of the relationship between service providers and services used, and thus provides an example of the amount of unneeded or unwanted incident information a subscriber could receive if the defined triggers are not taken into account. E.g. in the case of cloud B, it only uses a fraction of the services offered by cloud C and D. If cloud B was notified about all incidents at C and D, this would include a large amount of unneeded information relating to services not in use by cloud B. Additionally if cloud D has thousands of customers, the overhead of negotiating and defining which incident information to share would become noticeable. Therefore, it is expected to be better to use the hybrid approach, where the subscriber defines what he wants to be notified about and the provider additionally pushes all information required by law to the subscriber by using mandatory notifications that could be defined for each subscriber.

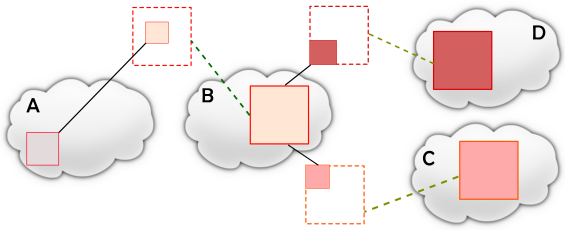

Figure 1: Data flow in supply chain. Each cloud represents a service provider. Each coloured square represents the services used by the subscriber. The coloured square inside each stippled square, represents the data or parts of the services actually used by the subscriber.

3.3 Propagation of Incidents

Incidents which affect one CSP could in turn affect other CSPs, and hence be propagated. Two approaches to representing propagated incidents have been considered. The first approach embeds the parent incident in an unchanged manner, allowing all recipients of a derived incident – an incident created based on information from another incident – to access all of the information received by an earlier link in the supply chain. At first glance, this seems like a valid approach and one that could result in better cooperation, faster response time and better incident handling as a result of access to more and better information. However, if all information from one provider is passed on to entities further down in the chain by a receiver, this would limit trust as the entity sending the incident information would not know who receives it. Given the potentially sensitive nature of information related to security incidents, there is a real possibility this would result in the system not being used or only superficial information being shared.

To allow the incident sender to be more in control of his incident information, a better approach is to only reference the parent incident when propagating down to a subscriber of a subscriber. In this way, only relevant information would propagate directly, through the actions of an entity, while there would still be a hard link to follow in order to establish exactly what happen during the incident handling. This could, e.g., be useful when auditing a provider or during criminal investigations.

4 SPECIFICATION

In order to allow cloud providers and cloud customers to exchange security incident information, we propose a standardised incident exchange format and subscription based Application Programming Interface (API). A standardised format allows for increased automation of incident response tasks, yet it does not require automation to be implemented anywhere. Only the interface provisioned by notification publishers need to be defined, while the implementation of the underlying functions can be up to the implementers, and can vary between different actors.

The specification we propose only deals with notifications between cloud providers and cloud customers, not notifications to end-users. If e.g. Dropbox uses Amazon S3 for storage of files, Dropbox is in the best position to know which end users to notify in the event of an incident happening at Amazon. Thus, this specification facilitates notification of end users by allowing incidents to propagate the cloud supply chain, while not directly handling the end user notification itself.

4.1 Notification Message Format

For the content of the incident notification messages, we propose a core format which supports manual incident handling as well, i.e. the case where the incident notification is read and acted upon by a human. Essential here is the capability to keep track of e.g. where an incident has occurred and how it may be correlated with other incidents, to make the notification system well suited for complex networks of services. In order to support automation, i.e. when a service could adapt itself during runtime to mitigate a threat (Gjære et al., 2014), we also make it possible to attach more structured and standardised formats to the message. A goal with our work is to keep the complexity to a minimum so that even small organisations/start-ups should be able to implement and utilise cloud incident notification tools. While some organisations only need manual handling, we retain the possibility to support more extensive formats needed for automation, at the same time facilitating specialisation and forward compatibility.

The fields included in the core format are based on a review of the RFC 5070 standard for an IODEF (Danyliw et al., 2007), the Federal Incident Notification Guidelines (US-CERT, 2014), the EU Commission Regulation No 611/2013 (EuropeanUnion, 2013), STIX (Barnum, 2012), and a shared mental model for incident response teams described by Flodeen et al. (Floodeen et al., 2013). Below, the format is presented in the JSON (ECMA International, 2013) notation with data type indications replacing the actual data:

```
{
    "id": UUID,
    "parent": {
        "id": UUID,
        "provider": STRING,
        "endpoint": URI
    },
    "type": {
        "id": UUID,
        "name": STRING,
        "description": STRING,
        "consequence": FLOAT,
    },
    "language": STRING,
    "status": STRING,
    "impact": FLOAT,
    "summary": STRING,
    "description": STRING,
    "occurrence_time": ISO 8601,
    "detection_time": ISO 8601,
    "liaison": {
        "id": UUID,
        "name": STRING,
        "email": EMAIL,
        "phone": STRING,
        "address": STRING,
        "zip": STRING,
        "city": STRING
    },
    "attachments": [
        {
            "format": STRING,
            "uri": URI,
        },
    ],
    "custom_fields": [
        {
            "id": UUID,
            "value": STRING,
            "type": {
                "id": UUID,
                "name": STRING,
                "description": STRING,
                "type": STRING, INT,
                    ↪ URI, JSON, etc.,
            }
        },
    ],
    "tlp": {
        "schema": STRING,
        "value": STRING,
        "fields": [
            {
                "field": STRING,
                "value": STRING
            },
        ]
    },
    "next_update": ISO 8601
}
```

The incident is associated with an incident type, and in addition its status is described (resolved/unresolved). The provider can give an estimate of the consequence of the incident (as a float number), and the meaning of this estimate is determined by the SLA. A short written summary of the incident, as well as possibly a more detailed description can be provided. In addition the (guessed) occurrence time of the incident as well as the time the incident was first detected are provided. The parent field provides traceability for propagated incidents, and the liaison field contains relevant contact information for this incident. By using the optional *tlp* field, a provider can instruct the receiving cloud customer on how to handle the received incident information. Given the difference between ENISA's Traffic Light Protocol (TLP) (ENISA, 2015) and the TLP described by US-CERT (US-CERT, 2015), it is possible to state which schema is used. The format also allows the sender to specify a different TLP value for specific fields. For any field not added specifically to the fields array, the main TLP value applies.

4.2 Custom Fields and Attachments

Our preliminary approach uses a small base format, with the ability to represent the most common information in a simple way, while providing a structured way of attaching other incident formats such as IODEF and STIX. In addition, the format supports *custom fields*, which allow the provider and the subscriber to agree upon extra information to be included in the base format without altering its base structure. This allows for two levels of implementation, as well as support incremental development of the highest level. *Level 1* would only implement the base format, and thus only be suitable for incident handling where there are humans acting on the incident reports. *Level 2* would implement different attachment formats, and thus be able to support more automation of incident handling. This allows for reuse of existing incident formats, specialization , incremental implementation , flexibility to support new formats, and the possibility to exchange evidence in the case of a forensic investigation.

Having a defined set of incident formats, agreed upon between the provider and subscriber through Service Level Agreements (SLAs), simplifies the implementation as a cloud customer would only have to support the attachment formats agreed upon with the cloud provider and the cloud provider only the formats agreed upon with its customers. Thus not all cloud providers and customers have to support all incident formats, but still gives flexibility to the provider and the subscriber in identifying the formats most suitable for their use case and apply those.

Custom fields allow providers and subscribers to agree on extra information to include in the base format without changing its structure. This is an easy way of exchanging a few extra values that the subscriber wants, but without the overhead of a large incident representation format. There are multiple ways custom fields could be implemented, including allowing any values to be included in any format, allowing only a predefined set of basic data values, and providing data building blocks that allow representation of anything in a structured way.

By allowing only basic data types, such as *String*, *Integer*, and *Boolean*, use of *custom fields* where *attachments* should have been used would become less likely, but still possible through abusing the *String* value field. Explicitly stating the type of the value in the field would make it easier for the subscriber to interpret the value. A potential problem with this approach is the reduced flexibility, though it might be argued that *attachments* should be used for any-

Table 1: REST interface for our solution.

Resource	URI	METHOD
Incident types	`/incidents/types`	GET POST* DELETE*
Incident type	`/incidents/types/{id}`	GET POST* DELETE*
Trigger types	`/incidents/types/{id}/triggers/types`	GET POST* DELETE*
Trigger type	`/triggers/types/{id}`	GET POST* DELETE*
Subscriptions	`/subscriptions`	GET POST
Subscription	`/subscription/{id}`	GET POST DELETE
Subscription incidents	`/subscriptions/{id}/incidents`	GET POST
Subscription incident	`/incidents/{id}`	GET POST DELETE
Notification triggers	`/incidents/{id}/triggers`	GET POST
Notification trigger	`/triggers/{id}`	GET POST DELETE
Notification validation	`/notifications/validate`	POST

thing more complex than including simple values. Thus only basic data types are allowed.

4.3 Prototype

The customers of a cloud provider may have varying preferences when it comes to which systems, services and incident types they are interested in notifications for, as well as which thresholds for severity these should operate in relation to. Supporting such individual preferences can be accommodated by publishers e.g. through offering customers a subscription mechanism. The individual provider will obviously need to have the final say as to what information their customers are allowed to receive, regardless of preferences. To avoid data leakage and enforce the principle of least privilege, access to the API is only provided through a secure channel, over HTTPS. Both senders and receivers are authenticated.

The API is implemented through Representational State Transfer (REST), which acts as an *adapter pattern* (Gamma et al., 1994) so the system is allowed to function on every platform as long as they implement the API and provide the mechanisms needed to support REST. Reducing coupling to a minimum increases flexibility, modifiability and portability, and makes it easier to adopt the solution also for established systems and solutions. A list of all endpoints along with their HTTPS method can be found in Table 1.

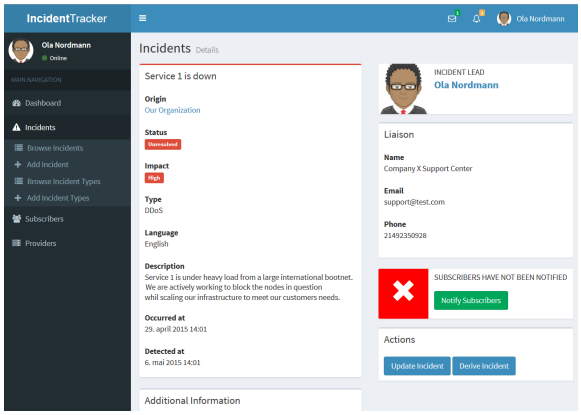

Figure 2: Incident Details from the prototype built upon the proposed solution.

Figure 2 shows a screen-shot from a prototype built around the concept and specification outlined in this paper. The prototyped graphical user interface presents a minimal way to manage the basic part of the incident format which is meant to be easy to understand for human incident handlers. As a subscriber, the dashboard allows you to browse through the received incident notifications. Currently the functionality to subscribe to incidents is not implemented in the user interface, but supported through the API. As a provider, the prototype has functionality for defining possible incident types which others can subscribe to, as well as composing new incident notifications and updates related to these. For those who are both a subscriber and a provider, the interface contains functionality for deriving a new message from a received notification, which shall retain links to the origin of the notification.

A link to the repository containing the prototype source code will be made available after the review.

5 INTERVIEWS

Catalyzed by the dashboard prototype, we have conducted two focused interviews with experienced incident handlers from two organisations to evaluate our approach. The participants both found the incident format to be mostly complete, but mentioned some special fields which could be useful in the base format. An indication on the time of next update was brought up in both interviews. An estimate on the expected time of correction, and the channel from which the incident report originally was received, was further mentioned as frequently useful information. In addition, adopting the Traffic Light Protocol (TLP) (US-CERT, 2015)(ENISA, 2015) was considered relevant for classifying confidentiality between parties, in terms of redistributing message contents. This was addressed in this iteration

of the prototype. A way to relay recommended actions to the subscriber was also mentioned as important. These suggestions should be examined further, and be included if there is a common need for them among incident handlers.

Given that other participants in the cloud delivery chain supported the REST interface and the incident format, the participants would find it useful for notifying customers. One participant also mentioned the possibility for such a solution to improve the amount and quality of reports to national Computer Emergency Response Teams (CERTs). What was missed from the interface was a way of requesting further information and reply to a report within the system. One participant suggested using a comment section available to all recipients of an incident, thus avoiding the sender having to respond to the same questions multiple times and allow collaboration between receiving incident handlers.

6 DISCUSSION

Here, we will discuss adoption, benefits for small and larger organizations, how our solution facilitates early decision making on sharing, comparing our solution to previous research and industry efforts and finally the need for further research.

Our main contribution is a simplified method of incident sharing, making it available for organizations of all sizes, while still allowing for implementing a large ecosystem with automation if so desired. While the proposed approach does not facilitate all cloud providers and customers in understanding all messages at any link in the chain, it ensures that all participants in the chain understand the information they are eligible to receive by requiring each link to be interoperable with its adjacent links.

In Section 2 it was pointed out that adoption of such an incident notification tool will likely depend on businesses expecting a positive economic impact from using it. The proposed solution is unlikely to contribute directly to a higher revenue stream, but might contribute indirectly. If a CSP, or any other organization adopting this solution, is diligent in sharing information about incidents, this could contribute to building an image of trustworthiness and professionalism. Such an image could in turn result in more customers, and thus increased revenue. A *Level 1* implementation, that is exchange of security incident information without any automation in incident handling, is unlikely to result in significantly reduced costs. However, the solution has been designed with implementation cost in mind, so the cost of adopting the solution should be quite low. The incremental nature of the solution allows implementers to gradually introduce more formats and also automation. As the implementation progresses into a *Level 2* implementation, with an increasing amount of automation, the reduced costs are expected to become noticeable. Metzger et al. (Metzger et al., 2011) claim that more than 85 % of abuse cases can be partly or fully automated, which in turn would free up resources allowing for reduced costs.

"Even the smallest organizations need to be able to share incident information with peers and partners in order to deal with many incidents effectively" (Cichonski et al., 2012). An important consideration taken when creating the proposed solution, was that it should be feasible for even small organizations to implement, utilize and receive value. The two level implementation allows for smaller organizations to reap a subset of the potential benefits, while larger organizations implementing full automation reaps the full set of potential benefits.

NIST (Cichonski et al., 2012) state that *"The incident response team should discuss information sharing with the organizations public affairs office, legal department, and management before an incident occurs to establish policies and procedures regarding information sharing."* Our proposed solution facilitates such decisions to be taken before incidents occur, as subscribers are able to subscribe to incident types made available to them by the CSP. Thus, the CSP needs to have decided beforehand which incident types each subscriber is allowed to subscribe to. In addition, each organization is free to decide how to implement the backend and can thus require a man in the loop, allowing for a second screening of incidents before they are sent to subscribers.

Comparing the approach presented in this paper with previous research efforts on sharing incident information, Cusick and Ma (Cusick and Ma, 2010) describe a solution conceptually similar to the one proposed here, but for internal use in the organization. Users might subscribe to be notified when events are created or changed, which has lead to improved communication around the incidents. The authors state that this feature alone made it worth their while to create a new process and implement new tooling. Our proposal takes this one step further, and allows other entities to be notified just as easily as the human subscribers.

As mentioned, both STIX and IODEF are excellent formats in their own respect. Still our main problem with both of them comes down to the generalized complexity and overwhelming scope. When a collection of large formats are supposed to represent everything, even the simple cases, this is likely to create some overhead in situations where what is represented does not fit the format. It seems like both have tried to counter this effect by making some fields optional, but Floodeen et al. (Floodeen et al., 2013) has found incident handlers in practice enters all information required by a system or a standard, rather than only the necessary information based on situation. Thus making fields optional will have a limited effect on which fields are used unless the tool strictly regulates which fields are presented to the incident handler.

Cusick and Ma had a different experience from that described by Floodeen et al. (Floodeen et al., 2013). Cusick and Ma found that engineers entered the minimum amount of information required and often just closed the ticket when it was handled without any information about steps taken to solve the incident. This problem will not be solved by the solution proposed here; if anything it will become more apparent and pressing. Compared to e.g. STIX the format presented in this paper is in danger of going too far in the opposite direction, which might result in overhead for the incident handler in deciding which information to enter – since the format does not necessarily ask for all the necessary information. Depending on the relationship between two parties exchanging incident information, it might be alarming for one party to never receive any information other than "we have an incident – it was solved". This might lead the

consuming organization to wonder if the incident has really been handled at all, and thus affect the trust between the two organizations. To solve this problem, organizations will probably need to consider the organizational culture as well as the use of pre-defined custom fields on the incident types which might serve as a way to mitigate this effect. The backend implementing this specification will also need to assist the incident handler in generating the relevant attachments when needed.

Still, there are many equally important challenges to tackle in order for sharing of security incident information to become common. Some of these are: trust, company culture, security in information sharing (like utilising broadcast encryption and signing of incident notifications), new mechanisms of collaboration, laws and SLAs. The interface we present here is simple enough to be used to facilitate further research into several of these areas. Our initial findings, from our limited number of interviews, indicate the need for a larger survey to be conducted on the fields included in the base format in order to learn more about its completeness and eventually which additional fields to include.

7 CONCLUSION

Incident information exchange is, at present, largely a manual exercise between two individuals who trust each other, via e.g. email, phone, or help desk systems. The same methods are prominent in both in-house and cloud chains – if ever incident information is exchanged at all. There are however legal requirements and contractual obligations which in some cases cements the need to exchange incident information in an effective manner.

The solution proposed in this paper is a first step in the direction of more effortless incident information exchange by means of formalistic and deterministic channels. Here we facilitate a direct connection between two parties in the supply chain, allowing for automated and/or manual handling of exchanged incident information. The proposal also allows for easy integration with existing systems and solutions, as well as use of internationally standardised message formats for possibly handling adaptation of services automatically during runtime.

ACKNOWLEDGEMENT

This work has been funded by the European Commission (grant nr. 317550).

REFERENCES

Bandyopadhyay, T., Mookerjee, V. S., and Rao, R. C. (2009). Why it managers don't go for cyber-insurance products. *Commun. ACM*, 52(11):68–73.

Barnum, S. (2012). Standardizing cyber threat intelligence information with the Structured Threat Information eXpression (STIX). Technical report, The MITRE Corporation.

Cain, P. and Jevans, D. (2010). Extensions to the IODEF-Document Class for Reporting Phishing. Technical report, IETF.

Cichonski, P., Millar, T., Grance, T., and Scarfone, K. (2012). Computer Security Incident Handling Guide: Recommendations of the National Institute of Standards and Technology, 800-61. Revision 2. Technical report, National Institute of Standards and Technology.

Cusick, J. J. and Ma, G. (2010). Creating an ITIL inspired Incident Management approach: Roots, response, and results. In *Network Operations and Management Symposium Workshops (NOMS Wksps), 2010 IEEE/IFIP*, pages 142–148. Ieee.

Danyliw, R., Meijer, J., and Demchenko, Y. (2007). The Incident Object Description Exchange Format.

ECMA International (2013). Ecma-404 the json data interchange standard.

ENISA (2015). Information disclosure.

EuropeanUnion (2013). Commission regulation (eu) no 611/2013 of 24 june 2013 on the measures applicable to the notification of personal data breaches under directive 2002/58/ec of the european parliament and of the council on privacy and electronic communications. Technical report.

Floodeen, R., Haller, J., and Tjaden, B. (2013). Identifying a shared mental model among incident responders. *Proceedings - 7th International Conference on IT Security Incident Management and IT Forensics, IMF 2013*, pages 15–25.

Frøystad, C. (2014). Exchange of security incident information in the context of cloud services. NTNU Minor Thesis Report.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design patterns: elements of reusable object-oriented software*. Pearson Education.

Gjære, E. A., Per, H., and Vilarinho, T. (2014). Notification Support Infrastructure for Self-Adapting Composite Services. In *DEPEND 2014, The Seventh International Conference on Dependability*, number c, pages 17–24, Lisbon, Portugal.

Greenleaf, G. (2012). The influence of European data privacy standards outside Europe: implications for globalization of Convention 108. *International Data Privacy Law*, 2(2):68–92.

Grobauer, B. and Schreck, T. (2010). Towards Incident Handling in the Cloud :. In *Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop*, pages 77–85, Chicago, Illinois, USA. ACM.

Horne, B. (2014). On Computer Security Incident Response Teams. *Security & Privacy, IEEE*, 12(October 2014):13–15.

Jaatun, M. G. and Tøndel, I. A. (2015). How much cloud can you handle? In *ARES 2015*.

Kalloniatis, C., Mouratidis, H., Vassilis, M., Islam, S., Gritzalis, S., and Kavakli, E. (2014). Towards the design of secure and privacy-oriented information systems in the cloud: Identifying the major concepts. *Computer Standards & Interfaces*, 36(4):759–775.

Metzger, S., Hommel, W., and Reiser, H. (2011). Integrated Security Incident Management – Concepts and Real-World Experiences. In *2011 Sixth International Conference on IT Security Incident Management and IT Forensics*, pages 107–121. Ieee.

Schneier, B. (2014). The Future of Incident Response. *Security & Privacy, IEEE*, 12(October):95–96.

The MITRE Corporation (2015). CybOX - Cyber Observable eXpression.

Torres, A. (2014). Incident Response : How to Fight Back A SANS Survey.

US-CERT (2014). Federal incident notification guidelines. Technical report.

US-CERT (2015). Traffic Light Protocol (TLP) Matrix and Frequently Asked Questions.