The missing piece in Threat Intelligence

Frank Denis
OVH
Email: frank.denis@corp.ovh.com

***Abstract*—Common systems for sharing intelligence on security threats have not been designed to efficiently process feedback from infrastructure providers. In order to fill this gap, we introduce DIP, a new description language to expose changes being made on a network that are relevant to security research and prevention.**

***Index Terms*—Information security, Information sharing**

I. Introduction

Information sharing has become increasingly important to reduce risk against security threats, and these information have proved to be useful for enterprise defense. However, from an infrastructure or service provider perspective, the current threat intelligence data and tools appear to be clearly insufficient.

Namely, confirming the relevance of a network-based indicator often requires private information that only network operators can access. Combining these private information with popular feeds of malicious indicators showed that a vast amount of these indicators were misleading or outdated. We also noted that security researchers did not have any convenient way to access information from providers about their infrastructure, how it is being used, and security-relevant changes being made.

As a proposal to solve this, we introduce DIP, a new description language designed for infrastructure and service providers. DIP allows them to share the unique knowledge they have about the networks they operate without actually disclosing any confidential data about their customers.

II. Network-based indicators are not permanent

An important observation is that network-based indicators require a time frame.

After having been confirmed as being part of a malicious campaign, the validity of some indicators is permanent. Malware samples and phishing emails belong to this category. They represent evidence whose malicious qualification doesn't depend on other features.

However, indicators based on network activities can only be trusted for a given time frame.

A. IP addresses

An IP address cannot be seen as unique and permanent identifier for an individual, a company, or an organization. There is a finite number of IP addresses, and their ownership is constantly changing.

Furthermore, it has become extremely common for IP addresses to be shared for multiple purposes and used by multiple customers.

1) Shared hosting: Web sites running on shared hosting platforms use a limited set of IP addresses to serve content for multiple customers.

2) Cloud computing services: Cloud computing services provide platforms whose resources are dynamically allocated. They are especially well suited for ephemeral tasks.

3) Content Delivery Networks: CDNs typically use a shared set of IP addresses to deliver content for many customers.

4) Dedicated addresses: Dedicated IP addresses and network blocks can be bought, sold, and, when rented from an infrastructure provider, reassigned to different customers.

B. Domain names

Domain names are not permanently assigned either. A domain name owner can only use a domain for as long as the related fees are being paid, and as long as it hasn't been taken down by the registrar. Unlike an IP address, a domain being deleted or assigned to a new owner is a publicly visible operation.

III. An experiment

Data shared about security threats are useful for manual investigation. But their main use remains to power automated or semi-automated reputation systems, eventually leveraged by security products and services for blocking possible attacks.

In the fight against cybercrime, infrastructure providers play a critical role. And information about malicious actors abusing their resources are essential for them to take action as quickly as possible.

In October 2015, we used the Combine [1] tool to retrieve the latest version of 37 public feeds as well as 4 commercial feeds tracking phishing, spam, malware and other suspicious activities.

For this experiment, we filtered out IP addresses having been listed for network scanning without any additional indicators.

Out of 518 incidents referencing the OVH network, we found 17 cases that were still relevant and active at the time the feeds were pulled, the remaining cases being:

- Servers that had been taken down since the incident actually happened
- Previously compromised servers that had been reinstalled
- IP addresses that had been reassigned to different customers
- IP addresses that had never been assigned
- Confirmed false positives

- Servers for which we were unable to find any evidence of malicious activity
- Tor exit nodes and proxies
- STUN servers and services returning information about HTTP clients and their IP addresses
- CDN and shared infrastructure

Using VirusTotal [2], we checked how many entries from these lists were flagged as malicious by common security products, while, from our perspective, not being actively serving malicious content any more.

VirusTotal had records for 27% of the involved IP addresses, and 20% of the domain names were still flagged as "malicious" or "suspicious" by at least one security product. While the actual number of domains and IP addresses being blocked by security products is believed to be vastly higher than the subset we checked, the ratio between indicators for live threats and the total number of indicators is likely to be similar.

IV. How long does an indicator remain relevant?

After having observed a network-based indicator being involved in malicious activities, determining how long this indicator should be considered valid is a mostly unsolved problem.

Evidence of repeated malicious activities involving a specific IP address should naturally lead to it being advertised as malicious, for its current and future activities, and no matter what its other uses are. If no more malicious activities are being observed, one still cannot confidently assume that the IP address has become totally benign.

Indeed, determining the optimal length of the observation period is a complex problem. First, the lack of evidence for known threat signatures doesn't imply that a server is not being involved in yet undetected threats. Secondly, even in the absence of activity, the IP can still be owned by the same malicious actors, and may be reused in future threats.

Applying an empirically defined time-to-live to network-based indicators is a common way to keep a balance between security and usability. However, this is clearly suboptimal, preventing access to legitimate and safe resources, while removing indicators that may still lead to an infection.

Companies and organizations providing the infrastructure for these threats could help solve this problem. In particular, they know when an IP address gets assigned to a customer, why and when the service is terminated, and when it gets reassigned to a different customer. They have a unique view on the network, servers and virtual resources they allocate to their customers.

V. Live threats vs indicators of compromise

Should a network-based threat indicator be permanently removed from intelligence feeds and databases after a confirmed takedown by its infrastructure provider?

There is no single answer to this question.

A command-and-control server being unintentionally contacted by a system remains a strong indicator that this system may have been infected, no matter what the current state of the C&C server is. A domain name known for having served a payload after having exploited a local vulnerability should also immediately trigger an alert, even if the payload is not accessible any more.

Even if they don't serve any malicious content any more, connections to these resources are indicators of compromise that remain permanently relevant.

On the other hand, compromised servers are commonly used as a starting point for an infection chain, and are not solid indicators of compromise after having been sanitized. However, the apparent absence of signs that a server is malicious is not a reliable indication that actions have been taken in order to restore the server security. The service provider, however, may have additional elements to confirm this.

Similarly, the response to a compromised domain name ("domain shadowing") used in an infection chain, is usually to block the whole domain name, as predicting the subdomains added for malicious purposes is rarely an option. Only the registrar can confirm whether the domain name is still at risk or if actions have been taken in order to secure it.

We therefore recommend threat intelligence sources and security services to make a clear distinction between live threats and indicators of compromises. Live threats must be blocked, as they present an immediate security risk, whereas indicators of compromise must trigger an alert, but may not prevent access to a service.

This, however, requires information from registrars and infrastructure providers.

VI. Engaging service providers

Service providers can help answer the following questions:

- Has the threat observed on this web site been removed? And when?
- More generally, what actions have been taken after an incident report?
- Is the IP address previously observed during an incident still being operated by the same actor?
- When was a server, a domain name or an IP assigned to a new customer?
- Is a given server, domain name or IP dedicated to a single user or shared by multiple unrelated customers?

These data can be extremely useful for security researchers, and to Law Enforcement Agencies. Reputation-based systems can take advantage of features derived from these data in order to improve their models. Security vendors can look at these data before accepting a whitelisting request from a customer.

Efficient mechanisms for security researchers to share information about threats do exist. But we are not aware of any automated and widespread mechanism for service providers to add information they possess to these knowledge bases.

Currently, complementary information from infrastructure providers:

- are only shared on request, after a threat was reported
- require one-on-one communications.
- cannot be automatically processed.

In order to improve this, we propose DIP, a description language that providers can easily deploy in order to transparently publish changes occurring on their network and the actions they take.

VII. DIP EVENTS DESCRIPTION

DIP is a minimal, machine-parseable language to describe events related to a specific infrastructure or service provider. These events are not observations, but actions having been performed as a response to an incident, as well as changes in associations between services and customers.

A major constraint in DIP is that is has to be able to expose changes without ever disclosing personal data about customers. Events must also be restricted to providing facts, and not opinions. While these events can be used by security company to build reputation systems, producers of DIP feeds should not weight in using this system.

Similarly, and unlike most information sharing systems, events do not include a confidence level. They are all assumed to have the same level of trust as the producer itself.

An event describes a single change, contains 7 mandatory properties, 1 type-dependent property, and 1 optional property.

id	event identifier	mandatory
time	timestamp	mandatory
type	resource type	mandatory
resource	resource identifier	mandatory
state	new state after the change	mandatory
source	source identifier	mandatory
depth	source depth	mandatory
owner	resource owner	type-dependent
related	related events and indicators	optional

A. Event identifier

Every event must include an identifier *id*, that cannot be reused for another event published by the same provider. This identifier is a Unicode string of any length, with no restrictions on the allowed set of characters.

B. Timestamp

A timestamp *time*, given as a Unix timestamp, is mandatory for all events. It has to represent the time an action was made, which can differ from the time the event is published.

C. Resource type

The presence of a property named *type* is essential for each event, and its value is a string that qualifies the type of the resource a change was made on.

- *domain*: the resource is a domain name. This represent an entire zone, and not a specific DNS record
- *nsrec*: a DNS record
- *vhost*: an entire set of services accessible via a specific host name
- *uri*: a complete URI
- *email*: an email address
- *ip*: an IP address
- *subnet*: an IP range

D. Resource identifier

A property named *resource* indicates what item of type *type* has been modified.

type	example
domain	example.com
nsrec	asd.example.com
vhost	example.com
uri	http://example.com/wp-includes/x.php
email	user@example.com
ip	192.0.2.42
subnet	192.0.2.0/24

E. New state after a change

The *state* property indicates the nature of a change.

- *assigned*: a new *owner* has been added for the *resource*, in addition to the possibly already existing list of owners
- *reserved*: the *resource* has been reserved by the provider for its own services.
- *unassigned*: a previous *owner* doesn't control the *resource* any more, but the *resource* can only be reassigned by the entity who previously assigned it. In particular, this applies to IP addresses and subnets, but not to domain names and virtual hosts
- *suspended*: the *resource* is still assigned to its previous list of owners, but was temporarily suspended by the service provider
- *resumed*: the *resource* is still assigned to its previous list of owners, and was put back online by the service provider after having been *suspended*
- *clean*: the service provider attests that no known security issues exist regarding the *resource*. This is used to report false positives
- *notified*: owners of *resource* have been notified by the service provider about a security issue
- *cleaned*: the service provider attests that known security issues regarding the *resource* have been addressed. An explicit transition to the *clean* state is not required.
- *deleted*: the *resource* doesn't exist any more or is not being used any more

F. Source identifier

source should be a globally unique identifier for the publisher of this particular event.

Similar to event identifiers, there are no restrictions on the length and on the allowed set of characters, but it must include the name of the company or organization.

G. Source depth

A provider is allowed to publish its own data, as well as to relay data from other companies or organizations it provides services for. This mechanism is detailed in the section VIII.

The value of the *depth* property for an event is initially set to 0, and must be incremented every time it gets relayed by an upstream DIP publisher.

H. resource owner

The resource owner represents an entity having full control over a resource, such as a domain name owner, a web site operator, or one of the accounts a server was assigned to.

The value of the *owner* property **must** change every time the actual owner of the resource changes.

In order to satisfy this, the value can be set according to the non-exhausting list of possible strategies:

1) Personal information identifying the owner
2) A unique account identifier, that doesn't disclose any personal information
3) A monotonically increasing counter
4) The output of a block cipher used in counter mode
5) A randomly chosen unique identifier

However, 1) goes against the DIP privacy goals, but can be used if full transparency is required, or if the data are only privately shared with trusted entities. 2) doesn't expose private information, but allows correlating different resources to the same entity, which may not be desirable. 3) is not recommended, as it gives a solid hint about the number of customers managed by the provider. 4) and 5) are therefore recommended for publicly available DIP data, as they do not disclose any information about the owner, and do not allow correlation.

These identifiers can be referenced in other information sharing systems.

I. Related

Finally, an optional *related* property can contain a list of unique internal or external identifiers for records related to the event.

STIX [3] identifiers are a natural fit for this property.

VIII. Aggregation and relaying

It is fairly common for infrastructure providers to delegate a part of the hardware and network resources they manage to resellers.

These resellers can publicly publish their own DIP data.

But a preferred alternative is to send the data to their upstream provider, which is going to aggregate its own data, as well as the data from its direct resellers, and eventually make them publicly accessible from a unique entry point.

In this scenario, a reseller exposes DIP events with a *depth* equal to 0 and its own *source* identifier. These events get consumed by the upstream provider, which increments their *depth* to 1 but retains the original *source* value.

These events, as well as those from other resellers, and events self-generated by the provider, are then made accessible under a unified interface.

The provider can review individual events published by its resellers.

In particular, it can confirm that a server was actually taken down or sanitized following a threat report. If the change or the action having been taken is confirmed, the provider must decrease the *depth* value before publishing the updated description of the event.

IX. A chain of trust

Published events cannot be ultimately trusted. For this reason, a DIP consumer has to explicitly choose the set of producers it is interested in.

This holds true at every level of the chain. A reseller consumes feeds from tier-2 resellers it trusts the feed information from. This reseller's infrastructure provider consumes and makes publicly available only the data from direct resellers it trusts the feeds from. Security vendors only select the feeds they trust, especially if they are automatically processed.

At any point in time, a consumer can stop reading or relaying a downstream feed if the information it contains doesn't appear to be correct or relevant.

X. Examples

DIP events are simple key/value and key/set pairs, and can use virtually any structured data representation.

However, for interoperability purposes, all implementations using this description language must be able to consume and publish data using the JSON format.

The following examples use this format.

A. An subnet owner change

```
{
  "id": "86be9a55762d316a3026c2836d044f5fc7",
  "time": 1446289736,
  "type": "subnet",
  "resource": "192.0.2.0/28",
  "state": "unassigned",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "owner": "ffe679bb831c95b67dc17819c63c509"
}
{
  "id": "a83dd0ccbffe39d071cc317ddf6e97f5c6",
  "time": 1446290241,
  "type": "subnet",
  "resource": "192.0.2.0/28",
  "state": "assigned",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "owner": "e7cf46a078fed4fafd0b5e3aff1448"
}
```

Note that the subnet doesn't transition to the *deleted* state. Even during the period it is not assigned to a customer, the subnet still exists, and can only be reassigned by the infrastructure provider.

B. A response to a phishing report


```
{
  "id": "7f71e4b6070f36e6c7e9c4b6f3d3bf1b",
  "time": 1446292030,
  "type": "uri",
  "resource": "http://phish.example.com/phish",
  "state": "suspended",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "related": ["example:Observable-160b1cd"]
}
{
  "id": "a2f95be4d1d7bcfa89d7248a82d9f111",
  "time": 1446292750,
  "type": "uri",
  "resource": "http://phish.example.com/phish",
  "state": "deleted",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "related": ["example:Observable-160b1cd"]
}
{
  "id": "a5193e54cd52837ed91e32008ccf41ac",
  "time": 1446292941,
  "type": "vhost",
  "resource": "example.com",
  "state": "deleted",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "related": ["example:Observable-160b1cd"]
}
{
  "id": "ba241029d241394997265a1a25aefc6",
  "time": 1446293713,
  "type": "domain",
  "resource": "example.com",
  "state": "deleted",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "related": ["example:Observable-160b1cd"]
}
```


In response to a phishing indicator, access to the phishing page was blocked, but everything else on example.com, including services on ”phish.example.com”, was left accessible.

After reviewing the customer account and web site content, the provider decided to take down the entire content. All the services using the ”example.com” were removed, followed by a deletion of the domain name itself.

Note that even though the phishing page might have been accessible using a given pattern, matching many URIs, what gets referenced in DIP events must be specific instances, and not the pattern itself, which is better left as a STIX object.

C. A response to a compromised server


```
{
  "id": "e4ff5e7d7a7f08e9800a3e25cb774534",
  "time": 1446293747,
  "type": "uri",
  "resource":
    "http://example.com/wp-includes/",
  "state": "cleaned",
  "source": "Reseller Inc",
  "depth": 1,
  "related": ["example:Observable-160b1cd"]
}
{
  "id": "d0752b60adb148ca0b3b4d2591874e2d",
  "time": 1446294279,
  "type": "uri",
  "resource":
    "http://example.com/wp-includes/",
  "state": "cleaned",
  "source": "Reseller Inc",
  "depth": 0,
  "related": ["example:Observable-160b1cd"]
}
{
  "id": "88aa3e3b1f22c616b1817981215e7d1",
  "time": 1446295013,
  "type": "vhost",
  "resource": "example.com",
  "state": "cleaned",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "related": ["example:Observable-160b1cd"]
}
```


A specific URI was reported as being malicious after a server compromise. The related service happened to be hosted by a reseller, who indicated that the reported threat had been addressed. That information was confirmed by the upstream provider, who also reviewed the whole ”example.com” web site and other services using that name, and labeled it as benign.

Note that the *clean* state would not be appropriate in that case, as the review was a response to an actual incident and not a false positive.

D. A response to a spam report


```
{
  "id": "3ad4e44a4306fb62b2df0ab7069c672a",
  "time": 1446295166,
  "type": "ip",
  "resource": "10.0.2.1",
  "state": "notified",
  "source": "Infrastructure Provider Corp",
  "depth": 0
  "related":
    ["http://spamtrap.example/4928"]
}
{
```

```
  "id": "fe1dcd3abfcd6b1655a026e60a05d0",
  "time": 1446295996,
  "type": "ip",
  "resource": "10.0.2.1",
  "state": "clean",
  "source": "Infrastructure Provider Corp",
  "depth": 0,
  "related":
    ["http://spamtrap.example/4928"]
}
```

In response to a spam case, the customer operating the source ip of the suspicious emails was notified. It doesn't imply that the problem was addressed, but attests that the report was not ignored. Note that an incident doesn't have to be directly reported to the provider for a DIP event to be logged. In this example, the case was found in an external feed.

After investigation, the report was confirmed to be a false positive. A new event with the *clean* state is thus generated.

XI. Implementation

DIP events can be published as feeds that are no different from other threat intelligence feeds, such as static lists of recent events, or using the TAXII project.

However, as these events represent incremental changes and not a global state, exposing them as a feed is not always convenient from an operational perspective. Answering the common questions "when was this IP address assigned to the current owner", "how many incidents were reported and addressed on this web site", or "is same subnet shared by many unrelated customers" can only be answered by replaying a sequence of events.

We therefore wrote ERIS [4], a reference implementation of an indexation engine and domain-specific query language for DIP events. ERIS provides an API allowing access to the complete history of a DIP resource.

ERIS a high-level API designed to be publicly accessible, that can rebuild the state of a resource for any date, or return a complete sequence of events for a given time frame. It also includes an indexation server that verifies, signs, merges and stores events in ArangoDB [5] although more storage engines can easily be plugged in.

The package includes a Python client library as well as a service to import DIP data to a CRITs [6] instance.

XII. Conclusion

Using DIP, Law Enforcement Agencies can have instant access to valuable information regarding resources linked to suspicious activities, including on past data.

Security researchers and SIEM operators can get instant feedback on reported threats and get more context to improve their models and products.

Service providers and incident responders can use DIP to save a tremendous amount of time by reducing the need for manual processing and one-on-one communication for communicating the actions they took.

Finally, end users get more visibility on the responsiveness of service providers regarding security threats.

DIP is extremely simple, yet fills a blind spot in the world of threat intelligence.

References

[1] *Combine*, MLSec project https://github.com/mlsecproject/combine
[2] *VirusTotal* http://www.virustotal.com
[3] *Structured Threat Information eXpression* http://stixproject.github.io/
[4] *ERIS* https://github.com/jedisct1/eris
[5] *ArangoDB* https://www.arangodb.com/
[6] *CRITs* https://crits.github.io/