Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?
Check out the list of upcoming events offering "Continuous Monitoring and Security Operations (Security 511)"
at http://www.giac.org/registration/gmon
Detection of Malicious Documents Utilizing XMP Identifiers

GIAC (GMON) Gold Certification

Author: Josiah Smith, josiahraysmith@gmail.com
Advisor: Bryan Simon

Accepted: 2020-August-02

Abstract

Modern digital documents are often composed of multiple other documents and images. Malware authors often produce malicious documents while reutilizing graphical assets or other components that can be uniquely identified with the Adobe Extensible Metadata Platform (XMP). XMP IDs define a standard for mapping asset relationships and can be utilized to track, pivot, and cluster malicious campaigns, identify new TTPs, and possibly provide attribution against adversaries.
1. Introduction

Phishing and malspam are delivery techniques heavily used to distribute malware, and Office documents still tend to be the malware filetype of choice (Verizon, p18, 2020). However, the execution of active content is necessary for the majority of techniques that are used to invoke the malicious intent of the author. Considering the necessity of the user to perform actions such as “enable editing” or “enable content”, the incorporation of a graphical element that compels the user to perform those actions is often included. These malware lures take on many different forms and appear in different file types and variations of written language. However, they often share a unique metadata attribute that can be used to track their existence and identify other potentially related instantiations.

These unique metadata attributes belong to the Extensible Metadata Framework and are called XMP Identifiers. This research aims to prove that malicious samples can be detected using these anchors. To adequately document a known directory of suspicious indicators, they will need to be derived from malicious files. The testing methodology for this process will generally follow the defined procedure:

1. Develop YARA rule to source files with XMP IDs and detection level > 10
2. Deploy YARA rule to VirusTotal Live Hunt operation.
3. Acquire known bad samples and download them to the lab environment.
4. Examine/Document XMP IDs from known harmful files.
5. Develop a new rule with suspicious XMP IDs.
6. Deploy back into live hunt rule for conditioning.
7. Deploy into continuous security operations.

XMP IDs can provide the security researcher or operations team the ability to track, pivot, and cluster malicious campaigns, identify new TTPs, and possibly offer attribution against adversaries.

2. What is XMP?

The Extensible Metadata Platform (XMP) provides a standard format for the creation, processing, and exchange of metadata within an assortment of applications. XMP was
developed by Adobe to solve the problem of incorporating a breadth of different metadata into various file formats while using a standard approach (Adobe, 2001). XMP allows tracking of parent-child relationships and different revisions and is most often stored in an XML format (Adobe, 2014). As described by Adobe (2001), there are three distinct properties within a packet, and together they define a managed asset.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DocumentID</td>
<td>xmpMM:DocumentID</td>
<td>Uniquely identifies an asset</td>
</tr>
<tr>
<td>InstanceID</td>
<td>xmpMM:InstanceID</td>
<td>Uniquely identifies a specific version of an asset</td>
</tr>
<tr>
<td>OriginalDocumentID</td>
<td>xmpMM:OriginalDocumentID</td>
<td>This value links an asset to its original source.</td>
</tr>
</tbody>
</table>

Figure 1: XMP Property Descriptions

While XMP IDs are not solely embedded within graphical assets and indeed can be used to identify other components of multiple file types, the re-utilization of the malware lure proves to be a useful anchor for malware discovery as confirmed by Amini and Remen (2019).

The recognition for the concept of utilizing XMP IDs to track malicious documents and their graphical assets is endowed to Michael Remen. When presented with the question, “What gave you the idea to start tracking XMP IDs?” He replied, “I noticed there was a lack of AV detection on this coercive type of malicious documents. While the images were benign, they were unique and being reused throughout multiple samples” (M. Remen, personal communication, 2020). The XMP IDs within the images have proved to be a useful anchor for detection.

3. Rule Generation

There are two different formats of XMP IDs that have been observed throughout this research. The Hash format and GUID format. The XMP definition (Adobe, 2008), states, “An ID should be guaranteed to be globally unique... Typically 128 or 144-bit numbers are used, encoded as hexadecimal strings.” The typical length of 128 or 144-bit numbers

Author Name, email@addressjosiahryasmith@gmail.com
infer the occurrence of 16 or 18-byte sequences within the ID and dictate the length that will be searched for within the regular expression. However, the predominant format is easily the 128-bit hash format based on the MD5 algorithm. The prevalence was dominant within this analysis of 9725 occurrences.

```
$ cat xmpidlist | grep xmp_ | cut -d: -f2 | distribution

Key|Ct   (Pct)     Histogram
$xmp_md5|9725 (100.00%) -----------------------------------------------
```

### 3.1. Regular Expressions

A regular expression, or regex, is a type of text pattern that can be utilized within a variety of different programming languages and applications. In order to develop a regex to match on an XMP-ID, it is imperative to understand some of the basics. While reviewing the regex quick reference provided by Rubular (2020), it appears this regex can be quite simple with character classes.

Concerning the XMP-ID in MD5 format, if a regex was developed to match on the instance ID `xmp.iid:B3D4F1219157E911B37B9950729CB11D`, the following regex can be used to identify it as well as other identifiers in this format. This regex was easily generated using the notation within the square brackets defined as a character class (Goyvaerts & Levithan, 2012, p. 34). A character class matches a single character out of a range of possible characters, which covers the different categories of IDs and allows the specification of the hash value.

```
xmp\.[dio]id[-_]\{32\}
```

![Figure 2: Regular Expression](image)
3.2. YARA Rule Development

YARA is a tool that was developed to help information security researchers identify and classify malware samples. Often proclaimed as the pattern matching swiss army knife (Van Impe, 2015), its practicality is a derivative of the author’s imagination and can be used in a variety of practical applications, from data-loss detection to deep-dive forensics. Each rule is comprised of a set of strings and a condition of Boolean expressions that determine its detection logic.

In order to develop the first YARA rule, the two potential formats for XMP identifiers must be taken into consideration. The strings $xmp_md5 and $xmp_guid are defined with a regex to account for the different identifier categories and possible values. With the intent to scale down the data and provide a defined constant within this research, only Microsoft Office documents will be examined. The approach is applied with the definition of the $magic string. A magic number is a sequence of numbers or bytes that are embedded at or near the beginning of a file and can be used to indicate the file format. According to Microsoft (2020), the byte sequence {D0 CF 11 E0 A1 B1 1A E1 00 00 00} defines a Microsoft Office document and can be attributed to 21 different file extensions (“File signature database,” n.d.).

The following derived rule is telling YARA that any file containing any of the strings defining the XMP IDs and that has the magic byte sequence for Office Documents, will trigger the signature.

```
rule Adobe_XMP_Identifier_Within_Office_Document
{
    meta:
        Description = “This signature identifies Adobe Extensible Metadata Platform (XMP) identifiers embedded within Office Documents.”

    strings:
        $xmp_md5 = /xmp\.\[dio\]id\[-: _\][a-f0-9]{32}/ nocase ascii wide
        $xmp_guid = /xmp\.\[dio\]id\[-: _\][a-f0-9]{36}/ nocase ascii wide
}
```

Author Name, email@addressjosiahraysmith@gmail.com
Detection of Malicious Documents Utilizing XMP Identifiers

$magic = \{ D0\ CF\ 11\ E0\ A1\ B1\ 1A\ E1\ 00\ 00\ 00 \}$

condition:
(any of ($xmp_*) and $magic in (0..1024))

Figure 3: Office Document with XMP ID YARA Rule

3.3. VirusTotal Enterprise

To effectively source unique indicators from a massive corpus of malicious, benign, and potentially undetected files, this research is utilizing VirusTotal (VT) Enterprise. VT Enterprise is comprised of a multitude of valuable services including: VT Intelligence, VT Hunting, VT Graph, and VT API. Of specific utility, identification, parsing, and the acquisition will be accomplished with the Hunting and CLI services.

A useful component of the Hunting service is established with Livehunt and its respective notifications. Livehunt occurs when a users’ YARA rules are uploaded and applied against all files sent to VirusTotal from around the world. The resulting near real-time notifications are provided and can be accessed programmatically or through the REST API.

3.3.1. Deploying Live Hunt Rules

In order to employ a YARA rule to a Livehunt operation, a premium account is required. Within the VT Hunting Solution, simply navigate to the Rulesets sub-tab and generate a new ruleset that will be comprised of the necessary YARA rule(s). Since the intent of the research is to find graphical assets relating to malicious documents, the YARA rule will be modified to detect samples that have an arbitrary consensus between an assortment of AV vendors. This approach can be completed using a file search modifier within the YARA rule. Specifically, the modifier positive:x filters the results according to the number of antivirus vendors that were detected upon scanning it within the VirusTotal platform (VT, nd). For the initial sourcing of XMP-IDs found within graphical assets in malicious documents, the arbitrary number of positive detections was set modestly at a count of greater than ten. The resulting YARA condition was modified to resemble the following condition.

Author Name, email@addressjosiahraysmith@gmail.com
Detection of Malicious Documents Utilizing XMP Identifiers

3.3.2. Explanation of results

After generating the Livehunt rule, it is necessary to wait until there are notifications relating to that ruleset. The notifications can be configured to alert via email or through the user interface. Figure 4 depicts one such notification. Of note, the FileName, SHA256 hash, Rule Name, and detection quantity is represented along with additional information. Another option to source results more quickly is with the RetroHunt capability. Retrohunt provides the ability to scan all the files sent to VirusTotal in the past 12 months with defined rulesets.

Figure 4: Livehunt Result

If Livehunt notifications are not producing enough content for the given ruleset, the Retrohunt capability is a compelling feature that will scan a volume of more than 420M files (~680TB worth of data) within a few hours (Retrohunt, n.d.). However, this method does not allow for the inclusion of search modifiers (such as positive count), and the matches are capped at 10,000 per job. Fortunately, there is an automated way to resolve this uncertainty by using the VT CLI tool. The first requirement is to derive only the sha256 hash values that are appended to the end of the name provided for the retrohunt job. The most effective way to parse these hashes out is with an extended grep command:

```bash
$ egrep -o [a-z0-9]{64} retrohunt_results > retrohunt_hashes
```

Now the resulting list of hashes can be scrutinized using the VT `file-report` and by defining the search modifier for positives in this script.

```bash
#!/bin/bash
read -p 'What are the minimum detections requested? ' detections
cat retrohunt_hashes | while read HASH ; do
   p=$(vt file-report ${HASH} | jq '.results.positives')
   if [ $p -ge $detections ] ; then
      # Do something with the hash
   fi
```

Author Name, email@addressjosiahraysmith@gmail.com
3.4. Generate List of Suspicious XMP IDs

After curating a listing of malicious Office documents that have an occurrence of an XMP ID, the next objective would be to generate a list of suspicious XMP IDs. This process of producing that list would entail downloading the corpus of files, parsing out the identifiers with YARA and other command-line tools, and then developing a successive YARA rule with the suspicious IDs either manually, or preferably programmatically.

The most effective way to download a large assortment of files from VirusTotal would be to utilize the VT CLI and script out the process. Within the Livehunt notifications for the particular ruleset, there is an option to export hashes. Using the list of hashes, the following Bash one-liner will loop through the list and download the individual files with the file-download argument (VirusTotal, 2020).

```
$ cat hashlist | while read HASH; do vt file-download $HASH; done
```

One method used to parse out the XMP IDs from the corpus of malicious documents would be to run YARA recursively with the -s (--print-strings) option and then grep, cut, and sort the output. Using the distribution tool, it is apparent that the XMP IDs frequently occur throughout the collection of files (Ellis, Stearns, & Vivero, 2020).

```
$ yara ../rules/Adobe_XMP_Identifier_Within_Office_Document.rule . -rs \
| grep xmp_ | cut -d ":" -f3-4 | distribution
```

<table>
<thead>
<tr>
<th>Key</th>
<th>Ct (Pct)</th>
<th>Histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmp.did:D8830B7209206811822AD4D0C71569CB</td>
<td>684 (0.14%)</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>xmp.iid:04801174072068118C14D51DCC23285E</td>
<td>624 (0.13%)</td>
<td>&quot;&quot;</td>
</tr>
</tbody>
</table>

Author Name, email@addressjosiahryasmith@gmail.com
Detection of Malicious Documents Utilizing XMP Identifiers

<table>
<thead>
<tr>
<th>XMP ID</th>
<th>Frequency</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>F77F11740720681180386AE8045B185</td>
<td>623</td>
<td>0.13%</td>
</tr>
<tr>
<td>058011740720681192B0E7E160E4461F</td>
<td>404</td>
<td>0.08%</td>
</tr>
<tr>
<td>088011740720681192B0E7E160E4461F</td>
<td>354</td>
<td>0.07%</td>
</tr>
<tr>
<td>048011740720681192B0E7E160E4461F</td>
<td>354</td>
<td>0.07%</td>
</tr>
<tr>
<td>FB7F11740720681197A5F036D900B89E</td>
<td>352</td>
<td>0.07%</td>
</tr>
<tr>
<td>A06C7ECD3B2068118083A844A3C9CD3</td>
<td>312</td>
<td>0.06%</td>
</tr>
</tbody>
</table>

Now that there is a collection of XMP IDs that have been derived from the malicious Office documents, a new YARA rule can be generated. While the strings could be incremented manually, it would most likely be a timely task. Instead, the utilization of this novel_rule_generator will quickly facilitate the job (King, 2020).

```python
#!/usr/bin/env python
# Script to build a Yara rule with an input file & one string per line
# Author: Rob King (@TheKingAdRob)

import re
import sys

YARA_TEMPLATE = """
    rule {rule_name}
    {
        strings:
        {string_defs}
        condition:
            any of them
    }
"""

if len(sys.argv) != 3:
    sys.exit("usage: %s RULE_NAME INPUT" % sys.argv[0])

strings = []
with open(sys.argv[2], "r") as fp:
    for number, line in enumerate(fp):
        strings.append("$s%d = /%s/" % (number, re.escape(line.strip())))
```

Author Name, email@addressjosiahraysmith@gmail.com
print(YARA_TEMPLATE.format(rule_name=sys.argv[1],
    string_defs="\n\t".join(strings)))

Figure 5: Rule Generation Script

To generate a new rule, begin by executing the script in Figure 5 while providing a name for the output rule and an input file containing the list of XMP IDs that were derived from the Malicious Office documents.

```
$ ./novel_rule_generator.py Suspicious_XMPID.rule xmpidlist

rule Suspicious_XMPID.rule
{
    strings:
    $s0 = /xmp\.did\:0180117407206811822A8F29126FB70E/
    $s1 = /xmp\.did\:02528A3E07B4E911A469955DC09DF99B/
    $s2 = /xmp\.did\:03801174072068118083F44BDB59DEDD/
    $s3 = /xmp\.did\:042493DF32C0E4118D9B8BA7F9641847/
    $s4 = /xmp\.did\:0880117407206811822A9F1610E837E9/
    $s5 = /xmp\.did\:08801174072068119231FAC1EE99744A/
    ...
```

Now that the YARA rule containing strings of the identified suspicious XMP IDs has been generated, the process can reiterate with the addition into the VirusTotal Hunting rulesets. After creating a new ruleset, the Livehunt notifications can be used aggregate samples containing graphical assets that were found within the malicious Office documents.

Additionally, the current rendition of the suspicious XMP ID rule can be applied within an organization’s continuous security monitoring environment, with consideration for the intention of this research is to identify files that are potentially malicious based on similarities found between them. Since the graphical assets that coerce users to enable active content are often reused, their reappearance in ensuing files that are undetected by AV solutions should be investigated. The continuous curation of the ruleset is also wise; add additional indicators as discovered and remove any indicators that are prevalent within benign instances.

Author Name, email@addressjosiahraysmith@gmail.com
4. Results and Analysis

4.1. Sample A

The first file that will be discussed is a Microsoft Excel Spreadsheet with the SHA-1 hash of 6568d702e83bf5d1067b8bfadfcac6e696a57b49. This file was chosen for examination based on the presence of an XMP ID that reoccurred six times within the sample set xmp.did:EB9F13B8B630E71191CDA99F6C9671D8. This suspicious ID was also found within the following documents identified by the SHA-256 hashes:

<table>
<thead>
<tr>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>193cb0caa096ce0b9c6ea2ac6875d1dbf12d4db1d630a100bd1c5fb74288bf3e</td>
</tr>
<tr>
<td>96d95668ee4138a14d8802b00fe76ae09b8fe7f27356288ab9e50a5346bcf988</td>
</tr>
<tr>
<td>98e4695eb06b12221f09956c4ee465ca5b50f20c0a5dc0550cad02d1d7131526</td>
</tr>
<tr>
<td>9e848ebc3af4449e73845e27238fb28ea7e7f9d4727da3eafb794ec6559d6f27b</td>
</tr>
<tr>
<td>c296f8c5522b03890d2c3681a19035781401f1094529089e405138eb25e9da72</td>
</tr>
<tr>
<td>e9f1dab486c5cb784f23d59f4b598bcea88dc4876f8246e63a8e4f033ebbf54</td>
</tr>
</tbody>
</table>

A more comprehensive listing of files containing this XMP ID can be acquired from InQuest Labs research portal at the following URL:
https://labs.inquest.net/dfi/search/ioc/xmpid/xmp.did%3ABEB05E267EBBE6118943A1A884F74A77

The ensuing image is the graphical asset utilized within this Office document (Yes, now this research paper includes suspicious images). The image invokes the social engineering effort to coerce the pitiable user into enabling the active content and executing the embedded logic. This particular instance is utilizing the re-birthed techniques of obfuscated Excel 4.0 macros. The additional analysis derives high confidence that this file is part of an invoice-themed malspam campaign and the resulting payload found at hxxps://merystol[.]xyz/6ng688x8 which was removed at the time of this research.

Author Name, email@addressjosiahraysmith@gmail.com
Detection of Malicious Documents Utilizing XMP Identifiers

Document created using the application not related to Microsoft Office.

For viewing/editing, perform the following steps:

Click Enable editing button from the yellow bar above

Once you have enabled editing, please click Enable content button from the yellow bar above

Figure 2: Coercive Lure A

While the pivot to additional files was shown using the XMP ID above, 13 identifiers were found to be associated with the image.

```
$ yara Adobe_XMP_Identifier carving-000.jpg -s | cut -d: -f 3-4 | \
  sort | uniq
xmp.did:2F3641AF08ABF655F001BDA26C84
xmp.did:32A1613310ABE611BF55F001BDA26C84
xmp.did:346765EC20ABE611BF55F001BDA26C84
xmp.did:3974A9586BBE6118943A1A884F74A77
xmp.did:696798C1B28E711A303AD291C1DD996
xmp.did:77CD87D3406BE611B390B0985243E8E
xmp.did:BE05E267EBE6118943A1A884F74A77
xmp.did:C4B05E267EBE6118943A1A884F74A77
xmp.did:EB9F13B8B630E71191CDA99F6C9671D8
xmp.iid:EB9F13B8B630E71191CDA99F6C9671D8
xmp.iid:EC9F13B8B630E71191CDA99F6C9671D8
xmp.iid:ED9F13B8B630E71191CDA99F6C9671D8
```

Author Name, email@addressjosiahaysmith@gmail.com
4.2. Sample B

The second file reviewed was a malicious Microsoft Word document with a SHA1 hash of 936252aa4f656eb5def659a00b9a818233d6933b. This graphical asset has the following XMP IDs associated with its presence. Pivoting of the first identifier, xmp.did:3BA1613310ABE611BF55F001BDA26C84, dozens of malicious documents were observed.

```
xmp.did:3BA1613310ABE611BF55F001BDA26C84
xmp.iid:306765EC20ABE611BF55F001BDA26C84
xmp.iid:3BA1613310ABE611BF55F001BDA26C84
xmp.iid:3CA1613310ABE611BF55F001BDA26C84
```

Figure 3: Coercive Lure B

1. Open the document in Microsoft Office. Previewing online is not available for protected documents
2. If this document was downloaded from your email, please click “Enable Editing” from the yellow bar above
3. Once you have enabled editing, please click “Enable Content” from the yellow bar above

Author Name, email@addressjosiahaysmith@gmail.com
4.3. Sample C

Examining the file 061269ff3fba696a1da5363f1723525e468e6ef presented the following XMP-IDs associated with the graphical lure that coerces execution.

```
$ yara Adobe_XMP_Identifier carving-000.jpg -s | grep xmp_ | cut -d: \ -f 3-4
xmp.did:03801174072068118083F44BDB59DEDD
xmp.did:08801174072068119231FA1EE9744A
xmp.did:4FD4E7F18A2068118527A7954BFDAD06
xmp.did:542CEC94152068119231FA1EE9744A
xmp.did:788A2A26D02511E7A0AEC87962BABB1D
xmp.did:CA0879F7C71211E89876B7EB71914955
xmp.did:D27578FF33206811871FD2CE7718D7F3
xmp.did:F77F117407206811A961EA0D9E399328
xmp.did:FB301D6D11206811871FD2CE7718D7F3
xmp.did:FC301D6D11206811871FD2CE7718D7F3
```

Figure 4: Coercive Lure C
While pivoting on xmp.did:542CEC94152068119231FAC1EE99744A, there were 14 related malicious files available.

5. InQuest Labs

While most data observed was sourced from VirusTotal, it is prudent to mention the capabilities of the open-source research portal, InQuest Labs (InQuest, 2020). Capable of ingesting malware at scale, samples are fed through a lightweight and less featured version of Deep File Inspection to extract embedded logic, semantic content, metadata, and IOCs such as URLs, domains, IPs, e-mails, and file names.

Currently, Microsoft and Open Office documents, spreadsheets, and presentations are available for search and download. In the future, there will be an expansion of the public data set to include Adobe PDF documents, Java / Flash applets, and scriptlets, such as PowerShell. Security researchers can search extracted layers and IOCs by keyword, download samples, and pivot between samples by heuristic detections and IOCs. InQuest Labs has documented thousands of suspicious XMP identifiers, and the user interface provides the ability to search through a vast set of samples. Furthermore, the pivot
functionality of IOCs provides for the correlation of samples with matching attributes like XMP IDs.

![Figure 5: Suspicious XMP Search](image)

6. Conclusion

Cyber threats have become commonplace throughout the world and have heightened the importance of security monitoring, threat hunting, and incident response. Considering the most prevalent attacks require some action from the target to be effective, detection and timely intervention are more critical than ever. One useful technique to detect threats and related campaigns is the examination and tracking of the XMP identifiers used within the malicious documents. Diving into a process to acquire, curate, and redeploy suspicious indicators has been examined. By leveraging XMP IDs for signature development, effective threat detection can be enveloped within an organization's security monitoring posture.

Author Name, email@addressjosiahsmith@gmail.com
References


File signature database:: D0cf11e0a1b11ae1 file signatures. (n.d.). File Signature Database:: https://www.filesignatures.net/index.php?page=search&search=D0CF11E0A1B11AE1&mode=SIG


Author Name, email@addressjosiahraysmith@gmail.com


### Appendix (SHA256 Hashes derived from XMP IDs)

#### Sample A: xmp.did:EB9F13B8B630E71191CDA99F6C9671D8

<table>
<thead>
<tr>
<th>HASHES STUDIED</th>
<th>SHA256 HASHES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample A</td>
<td>193cb0caa096cc0b9c6ea2ae6875d1dbf124db1d630a100bd1c5fb74288bf3e97f28eae5e1872a55d504eb527e1f2ece0f4742ea4e5926b39db68662f295f6af7433be001ccf8de859cb60f0bf50e043743b3a426f40323d915c9c40f9af1a3658d1e9c45ef564c3515e801635581a8a77cfef41fa4927f005699595c66c8f03e3b23661a6db77a28dabef5f23673e3c0c927f4205bbcb64a88a24f9eb53d98fb731105a96db1882a3e1d6975edd56e80f9ddccdb1232805d6dec5ca06ab20bf72bbb3c15f527a736b5778cd9f4b4b47be73b2329bd6ea86281db092f803b73f1c70e5dd94681b1fb21f512ed1ca4aafbf9faa1408777a7e78daa6559707def09f636552fb22b79fd9701cadcde955565b38df12f2be032ed9bac7e7f2fc9e361c0f03d5511c01ff8f9494ecfc721335328d6073d1640a7edc4ab5d2fb4206fa5c52a984ce661fddff1a9ed789129b6ba40ed228b924b66d5305ceceb1ce99efe03db92784aba345a3bf060a54859d2d2a8d64758315340f21d9321ebe33734ec9f2175c2b77319702bd261185167d4365be4278898115342e3729f7db246e98aa037cbf0d2ae041d359af086945c54c7c501354e9390764cf7ff7de045e8a91666781cf9482e144e6f68bbc0872b2b56c4ecae9c40f0df2f600cc0a6a910e62d714f01735b867708ec977100a0e368c10051d34ea36ff990d6d3ee1f1809f2a367df9c0762f6edcc03ae5f3f52d1ea3d11bffff15d1fa4e13820523e755b3b6bd22e4e49af3072b25358225f10e951111fe1c43202f5a47b78c23c63de182e2da30d8a56eae904d4f6f6cb28535cc725efc77b287008d4f7902ae88b1ae04009542e0720255e7a8974c636ff93b48f5fc9ec43b0cfb91bd701e5ef9034a4a7f7133f0c0bd935543cfff52d5dbdaa9f1eb53378f01190f87f291294db06a414c64e9df4a272522d868c81720204450ff4faebe5e9a373a0a771ae4404d8056f9098f0183b0de80e0ba491bb4a351a18e496f32a3b90c99d3858990a7d095a889e3b00c702d32116e83eca8f8cdaa243ed97f9ab01bbd56f8d83c79b06b66b32e6c7df92772088cb85234e75a9d34122</td>
</tr>
</tbody>
</table>

#### Sample B: xmp.did:3BA1613310ABE611BF55F001BDA26C84

<table>
<thead>
<tr>
<th>HASHES STUDIED</th>
<th>SHA256 HASHES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample B</td>
<td>c4e28ef5e0bdace4275f50c7f0e8c28fa19f086c6f63b31dbd6db2cb094708088ce886b08775eda705c6b9767aaba08765033e32b535cc575299891a9b15f206ff8e8ee84431e3c70f7b8dd9ec936204c0171ce9f1bca13e8ee6eb816017b5ed1e104b79b03e1d361e36b4228aa3364d112e9f2581aea2c2c2e995277f574df0375df34a059618b5c185df1ba3cc5ef02f36671558e3c10132039ddca6636ac757011d0cf2b66e68bd3e33044df0db0d88faeb2504ba9c9e6a848f64d60d6b1c5001ca9df02e4ed8c73d2dbcedf95ca02706697bb87b142af59c8d927e465c4d2521b8b9b373d22929381f73279b29d92fff73e62f5c5b6b9bd3615d655d1ddeb99366eae20c56894ea48284181059887a19046fa051772071d8a9527c7d96679b5880519a8bccc79d8a294b75b5396221374d9866bf146b484a60d3c31df85bf2293735b052d2db2e2ba1a2173fc77f524b790ddd8e47dad5f913c283993f922697711fdaf42b4b4e6035862687f944395c3061985e631ca37935be041162fc2b3dd21ea7c29005b709e5062ce376b3e64e019cac2165471823</td>
</tr>
</tbody>
</table>

Author Name, email@addressjosiahraysmith@gmail.com
Detection of Malicious Documents Utilizing XMP Identifiers

Sample C: xmp.did:542CEC94152068119231FAC1EE99744A

d5456147ea4579c7f22a48ae4f62d6432ff605707d9490a450702f0b307485d4
bb424b8755195e9d109ba274be45be48af847c36a2394b70e8025cb320c4e347
96d35357643338b355adeee447f84534966599e1c73c7b3049f232b2b18c71b9b
9d7dd89932a65ca8b3b3300434527b7f14b56a7175cb526f1368ddf2b39d50e
a47db516311d4647223114e2b54bd5a33fb6e156c71486ba42bb2c29f1c16db52
6f0b09444670d89ec825e151c95e522c60bd764906995371c25a0af516775c
637a2678016822c45a019b3764ca8d4f9b5d4ce64bdfa5e2f91be4f063d92
48c31d2f33fe11cb1e0135f4d763e4a37c27549444a9fa0d76ca49dd8857cc4e
0a55f500b5ba0e4c3ff9da59230302690be9542923aff4fb0af6eee8a2e593ddfd
e68c8166954d6baca0f5e61a64fdaadd8f5744e909e6944262c17997d42c0fe6
ec8ae56d47083be03044c12c2b4d719f38bcf1747ebbb908e4822dd9c6f4f24b
a2d83a3acc4ce11608bed8893eb71eba2292127e51b5edcecde4c4ee47a354c
6bad6e2bc1cf49e7cb20755584c5f924877cc54d27bda55028270a2f897c7
14869a8ab8ddc7e9fe39a4891236a1e87a5776190f43327f04db13cf02d2b37d

Author Name, email=addressjosiahrysmith@gmail.com
# Upcoming Training

<table>
<thead>
<tr>
<th>Training</th>
<th>Location</th>
<th>Dates</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Tokyo Autumn 2020</td>
<td>Tokyo, Japan</td>
<td>Oct 05, 2020 - Oct 17, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Rocky Mountain Fall 2020</td>
<td>Denver, CO</td>
<td>Nov 02, 2020 - Nov 07, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS London November 2020</td>
<td>London, UK</td>
<td>Nov 02, 2020 - Nov 07, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Gulf Region 2020</td>
<td>Dubai, United Arab Emirates</td>
<td>Nov 07, 2020 - Nov 26, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS San Diego Fall 2020</td>
<td>San Diego, CA</td>
<td>Nov 16, 2020 - Nov 21, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Amsterdam November 2020</td>
<td>Amsterdam, Netherlands</td>
<td>Nov 16, 2020 - Nov 21, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS London December 2020</td>
<td>London, UK</td>
<td>Dec 07, 2020 - Dec 12, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Cyber Defense Initiative 2020</td>
<td>Washington, DC</td>
<td>Dec 14, 2020 - Dec 19, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Security East 2021</td>
<td>New Orleans, LA</td>
<td>Jan 11, 2021 - Jan 16, 2021</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS OnDemand</td>
<td>Online</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
<tr>
<td>SANS SelfStudy</td>
<td>Books &amp; MP3s Only</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
</tbody>
</table>