I'm not robot
reCAPTCHA
Continue

How to read dmarc forensic report

A question people asked repeatedly in 2016 was whether or not their organization could deploy DMARC if they only used SPF at present. They knew the recommendation is to use both DKIM and SPF, and were concerned that their organizations couldn't benefit from DMARC without DKIM. The short answer is that you can use DMARC […] DMARC report processor Dmarcian.com has made a series of videos about DMARC and email authentication available on their website. There is no charge to view the nine videos, which are based on presentations Dmarcian has developed and offered over the past several years. You can find all nine videos here: There are good reasons why one domain might want to have the DMARC reports about its traffic sent to a mailbox in a different domain. However DMARC.org has recently received several aggregate reports for a few seemingly random domains with no connection to DMARC.org – which is potentially a form of abuse itself, and one […] Posted in Blog Tags aggregate reports, DMARC, reporting eco, the largest Internet industry association in Europe, released a paper in late March addressing DMARC and the data privacy laws of the EU. The short answer is that mailbox providers should feel free to implement DMARC filtering and send aggregate reports to domain owners, even under the very stringent data privacy laws of Germany. […] On April 15-16th multiple sources reported seeing an increase in the number of DMARC reports generated or received. Between two and three and a half times as many reports per day were observed, depending on the source. There was some hope there would be a public announcement during the RSA Conference in San Francisco last […] Posted in Blog Tags aggregate reports, DMARC, reporting, statistics Effective email authentication requires an organization to have clear control of its messaging operations, and it can take a lot of work to get there if you don't already have it. But many don't realize that deploying DMARC – before or while you're deploying DKIM and SPF – can offer a valuable external view of […] parsedmarc parsedmarc is a Python module and CLI utility for parsing DMARC reports. When used with Elasticsearch and Kibana (or Splunk), it works as a self-hosted open source alternative to commercial DMARC report processing services such as Agari Brand Protection, Dmarcian, OnDMARC, ProofPoint Email Fraud Defense, and Valimail. Parses draft and 1.0 standard aggregate/rua reports Parses forensic/failure/ruf reports Can parse reports from an inbox over IMAP Transparently handles gzip or zip compressed reports Consistent data structures Simple JSON and/or CSV output Optionally email the results Optionally send the results to Elasticsearch and/or Splunk, for use with premade dashboards Optionally send reports to Apache Kafka Demystifying DMARC - A complete guide to SPF, DKIM, and DMARC If you are looking for SPF and DMARC record validation and parsing, check out the sister project, checkdmarc. DMARC protects against domain spoofing, not lookalike domains. for open source lookalike domain monitoring, check out DomainAware. usage: parsedmarc [-h] [-c CONFIG_FILE] [--strip-attachment-payloads] [-o OUTPUT] [-n NAMESERVERS [NAMESERVERS ...]] [-t DNS_TIMEOUT] [--offline] [-s] [--debug] [--log-file LOG_FILE] [-v] [file_path [file_path ...]] Parses DMARC reports positional arguments: file_path one or more paths to aggregate or forensic report files or emails optional arguments: -h, --help show this help message and exit -c CONFIG_FILE, --config-file CONFIG_FILE a path to a configuration file (--silent implied) --strip-attachment-payloads remove attachment payloads from forensic report output -o OUTPUT, --output OUTPUT write output files to the given directory -n NAMESERVERS [NAMESERVERS ...], --nameservers NAMESERVERS [NAMESERVERS ...] nameservers to query -t DNS_TIMEOUT, --dns_timeout DNS_TIMEOUT number of seconds to wait for an answer from DNS (default: 2.0) --offline do not make online queries for geolocation or DNS -s, --silent only print errors and warnings --debug print debugging information --log-file LOG_FILE output logging to a file -v, --version show program's version number and exit Note In parsedmarc 6.0.0, most CLI options were moved to a configuration file, described below. parsedmarc can be configured by supplying the path to an INI file parsedmarc -c /etc/parsedmarc.ini For example # This is an example comment [general] save_aggregate = True save_forensic = True [imap] host = imap.example.com user = dmarcresports@example.com password = $uperSecure watch = True [elasticsearch] hosts = 127.0.0.1:9200 ssl = False [splunk_hec] url = token = HECTokenGoesHere index = email The full set of configuration options are: general save_aggregate - bool: Save aggregate report data to the Elasticsearch and/or Splunk save_forensic - bool: Save forensic report data to the Elasticsearch and/or Splunk strip_attachment_payloads - bool: Remove attachment payloads from results output - str: Directory to place JSON and CSV files in offline - bool: Do not use online queries for geolocation or DNS nameservers - str: A comma separated list of DNS resolvers (Default: Cloudflare's public resolvers) dns_timeout - float: DNS timeout period debug - bool: Print debugging messages silent - bool: Only print errors (Default: True) log_file - str: Write log messages to a file at this path n_procs - int: Number of process to run in parallel when parsing in CLI mode (Default: 1) chunk_size - int: Number of files to give to each process when running in parallel. Setting this to a number larger than one can improve performance when processing thousands of files imap host - str: The IMAP server hostname or IP address port - int: The IMAP server port (Default: 993) ssl - bool: Use an encrypted SSL/TLS connection (Default: True) skip_certificate_verification - bool: Skip certificate verification (not recommended) timeout - float: Timeout in seconds to wait for an IMAP operation to complete (Default: 30) max_retries - int: The maximum number of retries after a timeout user - str: The IMAP user password - str: The IMAP password (escape % with a second %) reports_folder - str: The IMAP folder where the incoming reports can be found (Default: INBOX) archive_folder - str: The IMAP folder to sort processed emails into (Default: Archive) watch - bool: Use the IMAP IDLE command to process messages as they arrive delete - bool: Delete messages after processing them, instead of archiving them test - bool: Do not move or delete messages elasticsearch hosts - str: A comma separated list of hostnames and ports or URLs (e.g. 127.0.0.1:9200 or :secret@localhost) Note Special characters in the username or password must be URL encoded. ssl - bool: Use an encrypted SSL/TLS connection (Default: True) user - str: Basic auth username password - str: Basic auth password cert_path - str: Path to a trusted certificates timeout - float: Timeout in seconds (Default: 60) index_suffix - str: A suffix to apply to the index names monthly_indexes - bool: Use monthly indexes instead of daily indexes number_of_shards - int: The number of shards to use when creating the index (Default: 1) number_of_replicas - int: The number of replicas to use when creating the index (Default: 1) splunk_hec url - str: The URL of the Splunk HTTP Events Collector (HEC) token - str: The HEC token index - str: The Splunk index to use skip_certificate_verification - bool: Skip certificate verification (not recommended) kafka hosts - str: A comma separated list of Kafka hosts user - str: The Kafka user passsword - str: The Kafka password ssl - bool: Use an encrypted SSL/TLS connection (Default: True) skip_certificate_verification - bool: Skip certificate verification (not recommended) aggregate_topic - str: The Kafka topic for aggregate reports forensic_topic - str: The Kafka topic for forensic reports smtp host - str: The SMTP hostname port - int: The SMTP port (Default: 25) ssl - bool: Require SSL/TLS instead of using STARTTLS skip_certificate_verification - bool: Skip certificate verification (not recommended) user - str: the SMTP username password - str: the SMTP password From - str: The From header to use in the email to - list: A list of email addresses to send to subject - str: The Subject header to use in the email (Default: parsedmarc report) attachment - str: The ZIP attachment filenames message - str: The email message (Default: Please see the attached parsedmarc report.) Warning save_aggregate and save_forensic are separate options because you may not want to save forensic reports (also known as failure reports) to your Elasticsearch instance, particularly if you are in a highly-regulated industry that handles sensitive data, such as healthcare or finance. If your legitimate outgoing email fails DMARC, it is possible that email may appear later in a forensic report. Forensic reports contain the original headers of an email that failed a DMARC check, and sometimes may also include the full message body, depending on the policy of the reporting organization. Most reporting organizations do not send forensic reports of any kind for privacy reasons. While aggregate DMARC reports are sent at least daily, it is normal to receive very few forensic reports. An alternative approach is to still collect forensic/failure/ruf reports in your DMARC inbox, but run parsedmarc with save_forensic = True manually on a separate IMAP folder (using the reports_folder option), after you have manually moved known samples you want to save to that folder (e.g. malicious samples and non-sensitive legitimate samples). Here are the results from parsing the example report from the dmarc.org wiki. It's actually an older draft of the the 1.0 report schema standardized in RFC 7480 Appendix C. This draft schema is still in wide use. parsedmarc produces consistent, normalized output, regardless of the report schema. { "xml_schema": "draft", "report_metadata": { "org_name": "acme.com", "org_email": "noreply-dmarc-support@acme.com", "org_extra_contact_info": " , "report_id": "939165199496411646 3", "begin_date": "2012-04-27 20:00:00", "end_date": "2012-04-28 19:59:59", "errors": [] } , "policy_published": { "domain": "example.com", "adkim": "r", "aspf": "r", "p": "none", "sp": "none", "pct": "100", "fo": "0" } , "records": [{ "source": { "ip_address": "72.150.241.94", "country": "US", "reverse_dns": "adsl-72-150-241-94.shv.bellsouth.net", "base_domain": "bellsouth.net" } , "count": 2, "alignment": { "spf": true, "dkim": false, "dmarc": true } , "policy_evaluated": { "disposition": "none", "dkim": "fail", "spf": "pass", "policy_override_reasons": [] } , "identifiers": { "header_from": "example.com", "envelope_from": "example.com", "envelope_to": null } , "auth_results": { "dkim": [{ "domain": "example.com", "selector": "none", "result": "fail" }] , "spf": [{ "domain": "example.com", "scope": "mfrom", "result": "pass" }] }] }] } xml_schema,org_name,org_email,org_extra_contact_info,report_id,begin_date,end_date,errors,domain,adkim,aspf,p,sp,pct,fo,source_ip_address,source_country,source_reverse_dns,source_base_domain,count,spf_aligned,dkim_aligned,dmarc_aligned,disposition,policy_override_reasons,policy_override_comments,envelope_from,header_from,envelope_to,dkim_domains,dkim_selectors,dkim_results,spf_domains,spf_scopes,spf_results draft,acme.com,noreply-dmarc-support@acme.com, 9391651994964116463,2012-04-27 20:00:00,2012-04-28 19:59:59,,example.com,r,r,r,none,none,100,0,72.150.241.94,US,adsl-72-150-241-94.shv.bellsouth.net,bellsouth.net,2,True,False,True,none,,,example.com,example.com.,,example.com,none,fail,example.com,mfrom,pass Thanks to Github user xennn for the anonymized forensic report email sample. { "feedback_type": "auth-failure", "user_agent": "Lua/1.0", "version": "1.0", "original_mail_from": "sharepoint@domain.de", "original_rcpt_to": "peter.pan@domain.de", "arrival_date": "Mon, 01 Oct 2018 11:20:27 +0200", "message_id": "", "authentication_results": "dmarc=fail (p=none, dis=none) header.from=domain.de", "delivery_result": "policy", "auth_failure": ["dmarc"], "reported_domain": "domain.de", "arrival_date_utc": "2018-10-01 09:20:27", "source": { "ip_address": "10.10.10.10", "country": null, "reverse_dns": null, "base_domain": null } , "authentication_mechanisms": [], "original_envelope_id": null, "dkim_domain": null, "sample_headers_only": false, "sample": "Received: from Servernameone.domain.local (Servernameone.domain.local [10.10.10.10])\tby mailrelay.de (mail.DOMAIN.de) with SMTP id 38.E7.30937.BD6E1BB5; Mon, 1 Oct 2018 11:20:27 +0200 (CEST)Date: 01 Oct 2018 11:20:27 +0200Message-ID: To: from: \"=?utf-8?B?SW50ZXJha3RpdmUgV2V0dGJld2VyYmVyLcOcYmVyc2ljaHQ=?=\" Subject: SubjectMIME-Version: 1.0X-Mailer: Microsoft SharePoint Foundation 2010Content-Type: text/html; charset=utf-8Content-Transfer-Encoding: quoted-printable", "parsed_sample": { "from": { "display_name": "Interaktive Wettbewerber-Übersicht", "address": "sharepoint@domain.de", "local": "sharepoint", "domain": "domain.de" } , "to_domains": ["domain.de"], "to": [{ "display_name": null, "address": "peter.pan@domain.de", "local": "peter.pan", "domain": "domain.de" }], "subject": "Subject", "timezone": "+2", "mime-version": "1.0", "date": "2018-10-01 09:20:27", "content-type": "text/html; charset=utf-8", "x-mailer": "Microsoft SharePoint Foundation 2010", "body": "", "received": [{ "from": "Servernameone.domain.local Servernameone.domain.local 10.10.10.10", "by": "mailrelay.de mail.DOMAIN.de", "with": "SMTP id 38.E7.30937.BD6E1BB5", "date": "Mon, 1 Oct 2018 11:20:27 +0200 CEST", "hop": 1, "date_utc": "2018-10-01 09:20:27", "delay": 0 }], "content-transfer-encoding": "quoted-printable", "message-id": "", "has_defects": false, "headers": { "Received": "from Servernameone.domain.local (Servernameone.domain.local [10.10.10.10])\tby mailrelay.de (mail.DOMAIN.de) with SMTP id 38.E7.30937.BD6E1BB5; Mon, 1 Oct 2018 11:20:27 +0200 (CEST)", "Date": "01 Oct 2018 11:20:27 +0200", "Message-ID": "", "To": "", "from": "\"Interaktive Wettbewerber-Übersicht\" ", "Subject": "Subject", "MIME-Version": "1.0", "X-Mailer": "Microsoft SharePoint Foundation 2010", "Content-Type": "text/html; charset=utf-8", "Content-Transfer-Encoding": "quoted-printable" }, "reply_to": [], "cc": [], "bcc": [], "attachments": [], "filename_safe_subject": "Subject" } } feedback_type,user_agent,version,original_envelope_id,original_mail_from,original_rcpt_to,arrival_date,arrival_date_utc,subject,message_id,authentication_results,dkim_domain,source_ip_address,source_country,source_reverse_dns,source_base_domain,delivery_result,auth_failure,reported_domain,authentication_mechanisms,sample_headers_only auth-failure,Lua/1.0,1.0,,sharepoint@domain.de,peter.pan@domain.de,"Mon, 01 Oct 2018 11:20:27 +0200",2018-10-01 09:20:27,Subject,,"dmarc=fail (p=none, dis=none) header.from=domain.de",,10.10.10.10,,,,policy,dmarc,domain.de,,False parsedmarc works with Python 3 only. Note If your system is behind a web proxy, you neeed to configure your system to use that proxy. To do this, edit /etc/environment and add your proxy details there, for example: http_proxy= :password@prox-server:3128 https_proxy= :password@prox-server:3128 ftp_proxy= :password@prox-server:3128 Or if no credentials are needed: http_proxy= https_proxy= ftp_proxy= This will set the the proxy up for use system-wide, including for parsedmarc. Warning If your mail server is Microsoft Exchange, ensure that it is patched to at least: Exchange Server 2010 Update Rollup 22 (KB4295699) Exchange Server 2013 Cumulative Update 21 (KB4099855) Exchange Server 2016 Cumulative Update 11 (KB4134118) On Debian 10 (Buster) or later, run: sudo apt-get install -y geoipupdate On Ubuntu systems run: sudo add-apt-repository ppa:maxmind/ppa sudo apt update sudo apt install -y geoipupdate On CentOS or RHEL systems, run: sudo yum install -y geoipupdate The latest builds for Linux, macOS, and Windows can be downloaded from the geoipupdate releases page on GitHub. On December 30th, 2019, MaxMind started requiring free accounts to access the free Geolite2 databases, in order to comply with various privacy regulations. Start by registering for a free GeoLite2 account, and signing in. Then, navigate the to the License Keys page under your account, and create a new license key for the version of geoipupdate that was installed. Warning The configuration file format is different for older (i.e. =3.1.1) versions of geoipupdate. Be sure to select the correct version for your system. Note To check the version of geoipupdate that is installed, run: You can use parsedmarc as the description for the key. Once you have generated a key, download the config pre-filled configuration file. This file should be saved at /etc/GeoIP.conf on Linux or macOS systems, or at %SystemDrive%\ProgramData\MaxMind\GeoIPUpdate\GeoIP.conf on Windows systems. Then run To download the databases for the first time. The GeoLite2 Country, City, and ASN databases are updated weekly, every Tuesday. geoipupdate can be run weekly by adding a cron job or scheduled task. More information about geoipupdate can be found at the MaxMind geoipupdate page. On Debian or Ubuntu systems, run: sudo apt-get install -y python3-pip On CentOS or RHEL systems, run: sudo yum install -y python34-setuptools python34-devel sudo easy_install-3.4 pip Python 3 installers for Windows and macOS can be found at sudo -H pip3 install -U parsedmarc Or, install the latest development release directly from GitHub: sudo -H pip3 install -U git+ Note On Windows, pip3 is pip, even with Python 3. So on Windows, substitute pip as an administrator in place of sudo pip3, in the above commands. For the best possible processing speed, consider using parsedmarc inside a pypy3 virtualenv. First, download the latest portable Linux version of pypy3. Extract it to /opt/pypy3 (sudo mkdir /opt if /opt does not exist), then create a symlink: wget tar -jxf pypy3.5-7.0.0-linux_x86_64-portable.tar.bz2 rm pypy3.5-6.0.0-linux_x86_64-portable.tar.bz2 sudo chown -R root:root pypy3.5-7.0.0-linux_x86_64-portable sudo mv pypy3.5-7.0.0-linux_x86_64-portable /opt/pypy3 sudo ln -s /opt/pypy3/bin/pypy3 /usr/local/bin/pypy3 Install virtualenv on your system: sudo apt-get install python3-pip sudo -H pip3 install -U virtualenv Uninstall any instance of parsedmarc that you may have installed globally sudo -H pip3 uninstall -y parsedmarc Next, create a pypy3 virtualenv for parsedmarc sudo mkdir /opt/venvs cd /opt/venvs sudo -H pip3 install -U virtualenv sudo virtualenv --download -p /usr/local/bin/pypy3 parsedmarc sudo -H /opt/venvs/parsedmarc/bin/pip3 install -U parsedmarc sudo ln -s /opt/venvs/parsedmarc/bin/parsedmarc /usr/local/bin/parsedmarc To upgrade parsedmarc inside the virtualenv, run: sudo -H /opt/venvs/parsedmarc/bin/pip3 install -U parsedmarc Or, install the latest development release directly from GitHub: sudo -H /opt/venvs/parsedmarc/bin/pip3 install -U git+ If you would like to be able to parse emails saved from Microsoft Outlook (i.e. OLE .msg files), install msgconvert: On Debian or Ubuntu systems, run: sudo apt-get install libemail-outlook-message-perl If you would like to test parsedmarc and another report processing solution at the same time, you can have up to two mailto URIs each in the rua and ruf tags in your DMARC record, separated by commas. Some organisations do not allow IMAP, and only support Exchange Web Services (EWS)/Outlook Web Access (OWA). In that case, Davmail will need to be set up as a local EWS/OWA IMAP gateway. It can even work where Modern Auth/multi-factor authentication is required. To do this, download the latest davmail-version.zip from Extract the zip using the unzip command. Install Java: sudo apt-get install default-jre-headless Configure Davmail by creating a davmail.properties file # DavMail settings, see for documentation ## # Basic settings # Server or workstation mode davmail.server=true # connection mode auto, EWS or WebDav davmail.enableEws=auto # base Exchange OWA or EWS url davmail.url= # Listener ports davmail.imapPort=1143 ## # Network settings # Network proxy settings davmail.enableProxy=false davmail.useSystemProxies=false davmail.proxyHost= davmail.proxyPort= davmail.proxyUser= davmail.proxyPassword= # proxy exclude list davmail.noProxyFor= # block remote connection to DavMail davmail.allowRemote=false # bind server sockets to the loopback address davmail.bindAddress=127.0.0.1 # disable SSL for specified listeners davmail.ssl.nosecureimap=true # Send keepalive character during large folder and messages download davmail.enableKeepalive=true # Message count limit on folder retrieval davmail.folderSizeLimit=0 ## # IMAP settings # Delete messages immediately on IMAP STORE \Deleted flag davmail.imapAutoExpunge=true # Enable IDLE support, set polling delay in minutes davmail.imapIdleDelay=1 # Always reply to IMAP RFC822.SIZE requests with Exchange approximate # message size for performance reasons davmail.imapAlwaysApproxMsgSize=true # Client connection timeout in seconds - default 300, 0 to disable davmail.clientSoTimeout=0 ## Use systemd to run davmail as a service. Create a system user sudo useradd davmail -r -s /bin/false Protect the davmail configuration file from prying eyes sudo chown root:davmail /opt/davmail/davmail.properties sudo chmod u=rw,g=r,o= /opt/davmail/davmail.properties Create the service configuration file sudo nano /etc/systemd/system/davmail.service [Unit] Description=DavMail gateway service Documentation= Wants=network-online.target After=syslog.target network.target [Service] ExecStart=/opt/davmail/davmail /opt/davmail/davmail.properties User=davmail Group=davmail Restart=always RestartSec=5m [Install] WantedBy=multi-user.target Then, enable the service sudo systemctl daemon-reload sudo systemctl enable parsedmarc.service sudo service davmail restart Note You must also run the above commands whenever you edit davmail.service. Warning Always restart the service every time you upgrade to a new version of davmail: sudo service davmail restart To check the status of the service, run: Note In the event of a crash, systemd will restart the service after 5 minutes, but the service davmail status command will only show the logs for the current process. To vew the logs for previous runs as well as the current process (newest to oldest), run: journalctl -u davmail.service -r Because you are interacting with DavMail server over the loopback (i.e. 127.0.0.1), add the following options to parsedmarc.ini config file: [imap] host=127.0.0.1 port=1143 ssl=False watch=True Note Splunk is also supported starting with parsedmarc 4.3.0 To set up visual dashboards of DMARC data, install Elasticsearch and Kibana. Note Elasticsearch and Kibana 6 or later are required On Debian/Ubuntu based systems, run: sudo apt-get install -y apt-transport-https wget -qO - | sudo apt-key add - echo "deb stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list sudo apt-get update sudo apt-get install -y default-jre-headless elasticsearch kibana For CentOS, RHEL, and other RPM systems, follow the Elastic RPM guides for Elasticsearch and Kibana. Warning The default JVM heap size for Elasticsearch is very small (1g), which will cause it to crash under a heavy load. To fix this, increase the minimum and maximum JVM heap sizes in /etc/elasticsearch/jvm.options to more reasonable levels, depending on your server's resources. Make sure the system has at least 2 GB more RAM then the assigned JVM heap size. Always set the minimum and maximum JVM heap sizes to the same value. For example, to set a 4 GB heap size, set See for more information. sudo systemctl daemon-reload sudo systemctl enable elasticsearch.service sudo systemctl enable kibana.service sudo service elasticsearch start sudo service kibana start Without the commercial X-Pack or ReadonlyREST products, Kibana does not have any authentication mechanism of its own. You can use nginx as a reverse proxy that provides basic authentication. sudo apt-get install -y nginx apache2-utils Or, on CentOS: sudo yum install -y nginx httpd-tools Create a directory to store the certificates and keys: To create a self-signed certificate, run: openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout kibana.key -out kibana.crt Or, to create a Certificate Signing Request (CSR) for a CA, run: openssl req -newkey rsa:4096-nodes -keyout kibana.key -out kibana.csr Fill in the prompts. Watch out for Common Name (e.g. server FQDN or YOUR domain name), which is the IP address or domain name that you will be hosting Kibana on. it is the most important field. If you generated a CSR, remove the CSR after you have your certs Move the keys into place and secure them: cd sudo mv ssl /etc/nginx sudo chown -R root:www-data /etc/nginx/ssl sudo chmod -R u=rX,g=rX,o= /etc/nginx/ssl Disable the default nginx configuration: sudo rm /etc/nginx/sites-enabled/default Create the web server configuration sudo nano /etc/nginx/sites-available/kibana server { listen 443 ssl http2; ssl_certificate /etc/nginx/ssl/kibana.crt; ssl_certificate_key /etc/nginx/ssl/kibana.key; ssl_session_timeout 1d; ssl_session_cache shared:SSL:50m; ssl_session_tickets off; # modern configuration. tweak to your needs. ssl_protocols TLSv1.2; ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256'; ssl_prefer_server_ciphers on; # Uncomment this next line if you are using a signed, trusted cert #add_header Strict-Transport-Security "max-age=63072000; includeSubdomains; preload"; add_header X-Frame-Options SAMEORIGIN; add_header X-Content-Type-Options nosniff; auth_basic "Login required"; auth_basic_user_file /etc/nginx/htpasswd; location / { proxy_pass ; proxy_set_header Host $host; proxy_set_header X-Real-IP $remote_addr; proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for; } } server { listen 80; return 301 https://$host$request_uri; } Enable the nginx configuration for Kibana: sudo ln -s /etc/nginx/sites-available/kibana /etc/nginx/sites-enabled/kibana Add a user to basic authentication: sudo htpasswd -c /etc/nginx/htpasswd exampleuser Where exampleuser is the name of the user you want to add. Secure the permissions of the httpasswd file: sudo chown root:www-data /etc/nginx/htpasswd sudo chmod u=rw,g=r,o= /etc/nginx/htpasswd Restart nginx: sudo service nginx restart Now that Elasticsearch is up and running, use parsedmarc to send data to it. Download (right click the link and click save as) export.ndjson. Import export.ndjson the Saved Objects tab of the management page of Kibana. It will give you the option to overwrite existing saved dashboards or visualizations, which could be used to restore them if you or someone else breaks them, as there are no permissions/access controls in Kibana without the commercial X-Pack. parsedmarc 5.0.0 makes some changes to the way data is indexed in Elasticsearch. if you are upgrading from a previous release of parsedmarc, you need to complete the following steps to replace the Kibana index patterns with versions that match the upgraded indexes: Login in to Kibana, and click on Management Under Kibana, click on Saved Objects Check the checkboxes for the dmarc_aggregate and dmarc_forensic index patterns Click Delete Click Delete on the conformation message Download (right click the link and click save as) the latest version of export.ndjson Import export.ndjson by clicking Import from the Kibana Saved Objects page Starting in version 5.0.0, parsedmarc stores data in a separate index for each day to make it easy to comply with records retention regulations such as GDPR. For fore information, check out the Elastic guide to managing time-based indexes efficiently. Starting in version 4.3.0 parsedmarc supports sending aggregate and/or forensic DMARC data to a Splunk HTTP Event collector (HEC). The project repository contains XML files for premade Splunk dashboards for aggregate and forensic DMARC reports. Copy and paste the contents of each file into a separate Splunk dashboard XML editor. Warning Change all occurrences of index="email" in the XML to match your own index name. The Splunk dashboards display the same content and layout as the Kibana dashboards, although the Kibana dashboards have slightly easier and more flexible filtering options. Use systemd to run parsedmarc as a service and process reports as they arrive. Create a system user sudo useradd parsedmarc -r -s /bin/false Protect the parsedmarc configuration file from prying eyes sudo chown root:parsedmarc /etc/parsedmarc.ini sudo chmod u=rw,g=r,o= /etc/parsedmarc.ini Create the service configuration file sudo nano /etc/systemd/system/parsedmarc.service [Unit] Description=parsedmarc mailbox watcher Documentation= Wants=network-online.target After=network.target network-online.target elasticsearch.service [Service] ExecStart=/usr/local/bin/parsedmarc -c /etc/parsedmarc.ini User=parsedmarc Group=parsedmarc Restart=always RestartSec=5m [Install] WantedBy=multi-user.target Then, enable the service sudo systemctl daemon-reload sudo systemctl enable parsedmarc.service sudo service parsedmarc restart Note You must also run the above commands whenever you edit parsedmarc.service. Warning Always restart the service every time you upgrade to a new version of parsedmarc: sudo service parsedmarc restart To check the status of the service, run: service parsedmarc status Note In the event of a crash, systemd will restart the service after 10 minutes, but the service parsedmarc status command will only show the logs for the current process. To vew the logs for previous runs as well as the current process (newest to oldest), run: journalctl -u parsedmarc.service -r The Kibana DMARC dashboards are a human-friendly way to understand the results from incoming DMARC reports. Note The default dashboard is DMARC Summary. To switch between dashboards, click on the Dashboard link in the left side menu of Kibana. As the name suggests, this dashboard is the best place to start reviewing your aggregate DMARC data. Across the top of the dashboard, three pie charts display the percentage of alignment pass/fail for SPF, DKIM, and DMARC. Clicking on any chart segment will filter for that value. Note Messages should not be considered malicious just because they failed to pass DMARC; especially if you have just started collecting data. It may be a legitimate service that needs SPF and DKIM configured correctly. Start by filtering the results to only show failed DKIM alignment. While DMARC passes if a message passes SPF or DKIM alignment, only DKIM alignment remains valid when a message is forwarded without changing the from address, which is often caused by a mailbox forwarding rule. This is because DKIM signatures are part of the message headers, whereas SPF relies on SMTP session headers. Underneath the pie charts. you can see graphs of DMARC passage and message disposition over time. Under the graphs you will find the most useful data tables on the dashboard. On the left, there is a list of organizations that are sending you DMARC reports. In the center, there is a list of sending servers grouped by the base domain in their reverse DNS. On the right, there is a list of email from domains, sorted by message volume. By hovering your mouse over a data table value and using the magnifying glass icons, you can filter on our filter out different values. Start by looking at the Message Sources by Reverse DNS table. Find a sender that you recognize, such as an email marketing service, hover over it, and click on the plus (+) magnifying glass icon, to add a filter that only shows results for that sender. Now, look at the Message From Header table to the right. That shows you the domains that a sender is sending as, which might tell you which brand/business is using a particular service. With that information, you can contact them and have them set up DKIM. Note If you have a lot of B2C customers, you may see a high volume of emails as your domains coming from consumer email services, such as Google/Gmail and Yahoo! This occurs when customers have mailbox rules in place that forward emails from an old account to a new account, which is why DKIM authentication is so important, as mentioned earlier. Similar patterns may be observed with businesses who send from reverse DNS addressees of parent, subsidiary, and outdated brands. Further down the dashboard, you can filter by source country or source IP address. Tables showing SPF and DKIM alignment details are located under the IP address table. Note Previously, the alignment tables were included in a separate dashboard called DMARC Alignment Failures. That dashboard has been consolidated into the DMARC Summary dashboard. To view failures only, use the pie chart. Any other filters work the same way. You can also add your own custom temporary filters by clicking on Add Filter at the upper right of the page. The DMARC Forensic Samples dashboard contains information on DMARC forensic reports (also known as failure reports or ruf reports). These reports contain samples of emails that have failed to pass DMARC. Note Most recipients do not send forensic/failure/ruf reports at all to avoid privacy leaks. Some recipients (notably Chinese webmail services) will only supply the headers of sample emails. Very few provide the entire email. DMARC ensures that SPF and DKM authentication mechanisms actually authenticate against the same domain that the end user sees. A message passes a DMARC check by passing DKIM or SPF, as long as the related indicators are also in alignment. DKIM SPF Passing The signature in the DKIM header is validated using a public key that is published as a DNS record of the domain name specified in the signature The mail server's IP address is listed in the SPF record of the domain in the SMTP envelope's mail from header Alignment The signing domain aligns with the domain in the message's from header The domain in the SMTP envelope's mail from header aligns with the domain in the message's from header Some vendors don't know about DMARC yet; ask about SPF and DKIM/email authentication. Check if they can send through your email relays instead of theirs. Do they really need to spoof your domain? Why not use the display name instead? Worst case, have that vendor send email as a specific subdomain of your domain (e.g. noreply@news.example.com), and then create separate SPF and DMARC records on news.example.com, and set p=none in that DMARC record. Warning Do not alter the p or sp values of the DMARC record on the Top-Level Domain (TLD) – that would leave you vulnerable to spoofing of your TLD and/or any subdomain. When you deploy DMARC on your domain, you might find that messages relayed by mailing lists are failing DMARC, most likely because the mailing list is spoofing your from address, and modifying the subject, footer, or other part of the message, thereby breaking the DKIM signature. Ideally, a mailing list should forward messages without altering the headers or body content at all. Joe Nelson does a fantastic job of explaining exactly what mailing lists should and shouldn't do to be fully DMARC compliant. Rather than repeat his fine work, here's a summary: Do Retain headers from the original message Add RFC 2369 List-Unsubscribe headers to outgoing messages, instead of adding unsubscribe links to the body List-Unsubscribe: < Add RFC 2919 List-Id headers instead of modifying the subject List-Id: Example Mailing List Modern mail clients and webmail services generate unsubscribe buttons based on these headers. Do not Remove or modify any existing headers from the original message, including From, Date, Subject, etc. Add to or remove content from the message body, including traditional disclaimers and unsubscribe footers In addition to complying with DMARC, this configuration ensures that Reply and Reply All actions work like they would with any email message. Reply replies to the message sender, and Reply All replies to the sender and the list. Even without a subject prefix or body footer, mailing list users can still tell that a message came from the mailing list, because the message was sent to the mailing list post address, and not their email address. Configuration steps for common mailing list platforms are listed below. Navigate to General Settings, and configure the settings below Setting Value subject_prefix from_is_list No first_strip_reply_to No reply_goes_to_list Poster include_rfc2369_headers Yes include_list_post_header Yes include_sender_header No Navigate to Non-digest options, and configure the settings below Setting Value msg_header msg_footer scrub_nondigest No Navigate to Privacy Options> Sending Filters, and configure the settings below Setting Value dmarc_moderation_action Accept dmarc_quarentine_moderation_action Yes dmarc_none_moderation_action Yes Navigate to Settings> List Identity Make Subject prefix blank. Navigate to Settings> Alter Messages Configure the settings below Setting Value Convert html to plaintext No Include RFC2369 headers Yes Include the list post header Yes Explicit reply-to address First strip replyo No Reply goes to list No munging Navigate to Settings> DMARC Mitigation Configure the settings below Setting Value DMARC mitigation action No DMARC mitigations DMARC mitigate unconditionally No Create a blank footer template for your mailing list to remove the message footer. Unfortunately, the Postorius mailing list admin UI will not allow you to create an empty template, so you'll have to create one using the system's command line instead, for example: touch var/templates/lists/list.example.com/en/list:member:regular:footer Where list.example.com the list ID, and en is the language. Then restart mailman core. If a mailing list must go against best practices and modify the message (e.g. to add a required legal footer), the mailing list administrator must configure the list to replace the From address of the message (also known as munging) with the address of the mailing list, so they no longer spoof email addresses with domains protected by DMARC. Configuration steps for common mailing list platforms are listed below. Navigate to Privacy Options> Sending Filters, and configure the settings below Setting Value dmarc_moderation_action Munge From dmarc_quarentine_moderation_action Yes dmarc_none_moderation_action Yes Note Message wrapping could be used as the DMARC mitigation action instead. In that case, the original message is added as an attachment to the mailing list message, but that could interfere with inbox searching, or mobile clients. On the other hand, replacing the From address might cause users to accidentally reply to the entire list, when they only intended to reply to the original sender. Choose the option that best fits your community. In the DMARC Mitigations tab of the Settings page, configure the settings below Setting Value DMARC mitigation action Replace From: with list address DMARC mitigate unconditionally No Note Message wrapping could be used as the DMARC mitigation action instead. In that case, the original message is added as an attachment to the mailing list message, but that could interfere with inbox searching, or mobile clients. On the other hand, replacing the From address might cause users to accidentally reply to the entire list, when they only intended to reply to the original sender. LISTSERV 16.0-2017a and higher will rewrite the From header for domains that enforce with a DMARC quarantine or reject policy. Some additional steps are needed for Linux hosts. A Python package for parsing DMARC reports exception parsedmarc.InvalidAggregateReport[source]¶ Raised when an invalid DMARC aggregate report is encountered exception parsedmarc.InvalidDMARCReport[source]¶ Raised when an invalid DMARC report is encountered exception parsedmarc.InvalidForensicReport[source]¶ Raised when an invalid DMARC forensic report is encountered exception parsedmarc.ParserError[source]¶ Raised whenever the parser fails for some reason parsedmarc.email_results(results, host, mail_from, mail_to, mail_cc=None, mail_bcc=None, port=0, require_encryption=False, verify=True, username=None, password=None, subject=None, attachment_filename=None, message=None)[source]¶ Emails parsing results as a zip file Parameters results (OrderedDict) – Parsing results host – Mail server hostname or IP address mail_from – The value of the message from header mail_to (list) – A list of addresses to mail to mail_cc (list) – A list of addresses to CC mail_bcc (list) – A list addresses to BCC port (int) – Port to use require_encryption (bool) – Require a secure connection from the start verify (bool) – verify the SSL/TLS certificate username (str) – An optional username password (str) – An optional password subject (str) – Overrides the default message subject attachment_filename (str) – Override the default attachment filename (str message) – Override the default plain text body parsedmarc.extract_xml(input_)[source]¶ Extracts xml from a zip or gzip file at the given path, file-like object, or bytes. Parameters input – A path to a file, a file like object, or bytes Returns The extracted XML Return type str parsedmarc.get_dmarc_reports_from_inbox(connection=None, host=None, user=None, password=None, port=None, ssl=True, verify=True, timeout=30, max_retries=4, reports_folder='INBOX', archive_folder='Archive', delete=False, test=False, offline=False, nameservers=None, dns_timeout=6.0, strip_attachment_payloads=False, results=None)[source]¶ Fetches and parses DMARC reports from an inbox Parameters connection – An IMAPClient connection to reuse host – The mail server hostname or IP address user – The mail server user password – The mail server password port – The mail server port ssl (bool) – Use SSL/TLS verify (bool) – Verify SSL/TLS certificate timeout (float) – IMAP timeout in seconds max_retries (int) – The maximum number of retries after a timeout reports_folder – The IMAP folder where reports can be found archive_folder – The folder to move processed mail to delete (bool) – Delete messages after processing them test (bool) – Do not move or delete messages after