z/OS MVS Diagnosis: Tools and Service Aids (GA32-0905-00)

# Chapter: Data Privacy for Diagnostics

[additional Section]

## Data Privacy for Diagnostics: OA58114 Enhancements

The Data Privacy for Diagnostics Analyzer is introduced by the solution for OA58114. The Data Privacy for Diagnostics Analyzer provides the facilities to scan and identify data within dumps that may be sensitive personal information (SPI). Because of the complexity of guidelines, requirements and SPI data identification, the Analyzer requires installations to tailor its privacy controls for whatever unique distinctions are necessary to filter out SPI from diagnostic data. Over redaction is possible which can negatively impact problem diagnosis. Most system areas are tagged as not having sensitive data, so it is possible for some SPI to escape redaction. At its core is an application which runs via batch jobs. Those jobs may be tailored through an IPCS dialog, or manually managed by the installation. The details for setup and execution will be found within the Interactive Problem Control System (IPCS) framework which consists of documentation within the IPCS Customization, IPCS Commands and IPCS User's Guide publications [note: these are general book URL links].

# z/OS MVS IPCS Customization (SA23-1383-00)

## Topic: IPCS Installation Package

## SubTopic: Customizing the IPCS Installation Package

[Add a new Topic:] Customizing Data Privacy for Diagnostics

The Data Privacy for Diagnostics Analyzer provides the facilities to scan and identify data within dumps that may be sensitive personal information (SPI). The Data Privacy for Diagnostics Analyzer runs via batch jobs and utilizes the zFS file system to retain all of its required input, and as a repository for its reports. Required inputs include dictionaries used to identify SPI. Reports can be generated to help a user understand what caused pages to be flagged as containing sensitive data. Users will be able to provide feedback by updating this information in the file system and running the feedback analysis tool to improve the SPI data detection/analysis.

In order to use the Analyzer, some initial set up must be performed. To help with this set up, a sample batch job has been provided. The batch job will create and initialize the file system, mount it to the desired mount point, and run an

initialization shell script. See 'SYS1.SAMPLIB(BLSDPJIN)' for instructions on how to modify the sample batch job example to run it on a different system(s).

One consideration that should be given is to the access control for this file system. Some of the sub-directories may contain sensitive data that has been extracted from dumps, or data that has been ingested by your customization for dump analysis. This data may be in reports, in files after feedback has been given and data has been ingested. Therefore, you want to ensure that only intended personnel have access to these folders. One option for you is to use FSACCESS control user access to this data. See "z/OS Security Server RACF Security Administrator's Guide" for more information on using FSACCESS to control access to file systems.

Once this initial set up is complete, a user will want to ensure that the new file system is always mounted on the systems where the analysis will be run. The user may choose to update the appropriate BPXPRMxx SYS1.PARMLIB members to ensure that the mount processing occurs.

*The Data Privacy for Diagnostics Analyzer File System*

The set up job BLSDPJIN performs the following steps:

- Creates the file system
- Formats the file system
- Creates the home directory
- Mounts the file system to the home directory
- Runs the initialization shell script (blsdpdp.sh, see NOTE in Data Privacy for Diagnostics Analyzer Directory Maintenance section)

The file system has a required folder structure. The following describes these sub-directories and their contents:

- /<directory>/knowledgebase : This folder is used to store the ingested knowledge and user feedback.
    - /<directory>/knowledgebase/ingested/ : This folder stores the ingested knowledge such as user provided dictionaries and regular expressions, and is populated by the INGEST function.
    - /<directory>/knowledgebase/feedback/ :This folder stores the processed user feedback and is populated by the FEEDBACK function.
- /<directory>/configuration : This folder stores the configuration which is used for various operations carried out by Data Privacy for Diagnostics Analyzer. This folder will contain the following configuration files (which correspond to the ANALYZE, INGEST, and EXTRACT modes of operations)

    - /<directory>/configuration/analysis_config.json : This file contains configuration about the sensitivity analysis to be carried out on dump. It allows customizing which built-in identifiers and ingested information that should be used for analysis. It also allows you to customize which combination of identifiers should be present together in order for data to be considered sensitive. See the z/OS MVS IPCS User's Guide for additional information about the analysis_config.json file including parameters and examples.
    - /<directory>/configuration/extract_config.json : This file contains configuration about identifiers which are to be extracted to a file. It allows the user to display the current pattern or dictionary associated with a built-in or customer identifier that is available for the ANALYZE function in determining which data is to be marked as sensitive by the Analyzer. See the z/OS MVS IPCS User's Guide for additional information about the extract_config.json file including parameters and examples.
    - /<directory>/configuration/ingestion_config.json : This file contains configuration about user provided data to be ingested. The ingested data is then available for use in future ANALYZE runs. See the z/OS MVS IPCS User's Guide for additional information about the ingestion_config.json file including parameters and examples.
- /<directory>/reports : This folder is used to store reports generated by Data Privacy for Diagnostics Analyzer. A subdirectory is created for each dump on which ANALYZE processing is requested. The following folder structure is generated for each dump:
    - /<-directory>/reports/<dump-name-1>/ : This folder stores reports from each invocation of DPfD ANALYZE.
        - /<-directory>/reports/<dump-name-1>/<timestamp-of-DPfD-ANALYZE-invocation>/ : Stores reports of a single ANALYZE invocation. It contains the following files:
            - ../concise_sensitive_report_<i> : This file is generated by each thread spawned by ANALYZE to process the dump.
            - ../sensitive_token_log_<i> : This file is generated by each thread spawned by ANALYZE to containing all of the sensitive tokens identified in the dump . These files are generated by each thread spawned by the ANALYZE function if the value of the SENSITIVE REPORT field is Y on the IPCS ANALYZE panel or if the log_sensitive_tokens value is set to TRUE in the BLSJDPA JCL that invokes the ANALYZE function.

- ../sensitive_tokens : This file contains all the sensitive tokens identified in the dump along with their count. This file is generated when REPORT is requested. This file can be modified to provide feedback about tokens which are incorrectly marked as sensitive for a subsequent FEEDBACK run.
- ../non_sensitive_tokens : This file contains all the non-sensitive tokens identified in the dump along with their count. This file is generated when REPORT is requested after ANALYZE that requested token level redaction. Token level redaction can be requested by specifying the value N for the ALLOW PAGE LEVEL option on the ANALYZE IPCS panel or by specifying the value 2 for the analysis_mode option in the BLSJDPA JCL. This file can be modified to provide feedback about tokens which are incorrectly marked as non-sensitive.

*Data Privacy for Diagnostics* Analyzer *Directory Maintenance*

As more and more dumps are run through the various Data Privacy for Diagnostics Analyzer functions, the file system usage will grow. You should periodically determine if older dump analysis directories are no longer required, and remove those sub-directories and their contents.

NOTE: The blsdpdp.sh initialization shell script that runs at the end of the BLSDPJIN setup job will not update the analysis_config.json, ingestion_config.json and extract_config.json files in <directory>/configuration if those files are present in that directory when the script is run. To obtain the latest changes, you can preserve the contents of your current configuration files by renaming them or simply delete them if not needed. Then run the shell script to create newly updated versions of those configuration files. Lastly, merge the installation's changes from the renamed files into the resulting files as the Analyzer will only use the configuration files with the names analysis_config.json, ingestion_config.json and extract_config.json

z/OS MVS IPCS User's Guide (SA23-1384-00)

Topic: Using IPCS functions

Update existing section: Using the IPCS Dialog -> IPCS Primary Option Menu -> Option 5 - Submit

https://www.ibm.com/support/knowledgecenter/SSLTBW_2.4.0/com.ibm.zos.v2r4.ieac600/blsgbkd.htm

Update Figure 1 to add Option 6 as such:

Figure 1. IPCS MVS Dump Batch Job Option Menu Panel

```
 ---------------------- IPCS MVS DUMP BATCH JOB OPTION MENU ------
-------------
 OPTION ===> _

********************
    1SADUMP      - Prepare stand-alone dump for analysis * USERID
- USER1
    2SVCDUMP     - Prepare SVC dump for analysis         * DATE
- 95/05/17
    3SYSMDUMP    - Prepare SYSMDUMP for analysis         * JULIAN
- 95.137
    4SUPPLEMENT - Perform supplementary dump analysis    * TIME
- 16:45
    5EREP        - Process software data using EREP      * PREFIX
- USER1
    6DPfD        - Data Privacy for Diagnostics          *
TERMINAL- 3278
                                                         * PF
KEYS - 24
 JOB STATEMENT INFORMATION: (Verify before proceeding)   *
********************

   ===> //USER1 JOB ACCT57,'IBM PSR',NOTIFY=USER1,
   ===> // MSGCLASS=A,MSGLEVEL=(2,1)
   ===>
   ===>
   ===>
   ===>

Enter END to end batch job processing.
```

Please add the new option 6 Data Privacy for Diagnostics following after 5 EREP in the description section:

- 5 EREP

Use the EREP option to request the Environmental Record Editing and Printing Program (EREP) to process logrec records. EREP places the results in a SYSOUT data set, which you can read online.
...

- 6 Data Privacy for Diagnostics.
  Use this option to post-process SVC, stand-alone, or SLIP dumps taken on z15 or later processors in order to redact pages that have been tagged as being sensitive by the applications that created the pages, as well as untagged pages that will be scanned and detected as containing sensitive data per the Data Privacy for Diagnostics Analyzer. For additional information, please see the topic Using Data Privacy for Diagnostics in the z/OS MVS IPCS User's Guide.

[Add a new Topic:] Using Data Privacy for Diagnostics Analyzer

*Data Privacy for Diagnostics* Analyzer provides the capability to post process SVC, stand-alone dumps, or SLIP dumps taken on z15 or later processors in order to redact pages that have been tagged as being sensitive by the applications that created the pages, as well as untagged pages that will be scanned and detected as containing sensitive data per the Data Privacy for Diagnostics Analyzer. This redacted version of the original dump is written to a new dump dataset without modifying the original dump dataset. Retain both dumps for as long as it takes to diagnose the reported problem. Additional processing is required for stand-alone dumps which contain captured dumps. If the captured dumps are required by vendors, the dumps must first be extracted (IPCS COPYCAPD) from the original stand-alone dump, then processed separately. Redacted stand-alone dumps will not contain the captured dumps.

The following functions are being provided:

- REDACT: You may redact any data tagged as sensitive=yes without further analysis. NOTE: You cannot perform the ANALYZE function on a dump that has already been redacted via this process. You can request this processing using either:
  - IPCS option 5.6, specifying the ANALYZE function and BYPASS DP ANALYSIS=Y
  - Use sample job SYS1.SAMPLIB(BLSJDPFD)
- ANALYZE: Redact any pages tagged sensitive by the applications that own that data, as well as any untagged pages scanned and detected as containing sensitive data. NOTE: You cannot perform the ANALYZE function on a dump

that has already been redacted via this process. You can request this processing using either:
    - IPCS option 5.6, specifying the ANALYZE function and BYPASS DP ANALYSIS=N
    - Use sample job SYS1.SAMPLIB(BLSJDPA).
- REPORT: You may create human readable reports for a dump that has been processed by the Data Privacy for Diagnostics Analyzer. These reports, once created, are in the <directory>/reports/<dump-name>/<run-number> directory in the file system used for ~~DPA~~ Data Privacy for Diagnostics Analyzer processing. You can request this processing using either:
    - IPCS option 5.6, specifying the REPORT function.
    - Use sample job SYS1.SAMPLIB(BLSJDPR).
- FEEDBACK: You may provide feedback for a dump that has been processed by the Analyzer. After looking through the reports and understanding the pages that have or have not been flagged as sensitive, you can provide feedback to help the Data Privacy for Diagnostics Analyzer improve its sensitive data detection. More information is covered on providing feedback later in this chapter. After updating configuration files and indicating what tagging can be improved, you can request this processing using either:
    - IPCS option 5.6, specifying the FEEDBACK function
    - Use sample job SYS1.SAMPLIB(BLSJDPF).
- INGEST: You may ingest data to help the Data Privacy for Diagnostics Analyzer determine what sensitive data exists in your environment. Data can be ingested from dictionaries, databases or other sources. This data is added to the knowledge base information and will be used in future analysis runs. More information is covered on providing ingested data later in this chapter. After updating configuration files and indicating what tagging can be improved, you can request this processing using either:
    - IPCS option 5.6, specifying the INGEST function
    - Use sample job SYS1.SAMPLIB(BLSJDPI).
- EXTRACT: You may extract any built-in or custom identifiers from the Analyzer to a file so that the user may see the exact criteria for determining the sensitivity of the data via the ANALYZE function. The output file will contain either the pattern or entire dictionary depending on the type of identifier to assist in ensuring that the Data Privacy for Diagnostics Analyzer is correctly marking data as sensitive or non-sensitive. More information is covered on extracting identifiers later in this chapter. After updating configuration files and indicating which identifiers can be written to a file, you can request this processing using either:
    - IPCS option 5.6, specifying the EXTRACT function
    - Use sample job SYS1.SAMPLIB(BLSJDPX).

Generally, you will want to start by performing the ANALYZE function on a dump. Remember that this function only works on dumps captured on a z15 or later processor. After creating the redacted version of the dump, you will want to check the dump to understand what has been redacted. Reports are available to help you understand why pages have been redacted. You can look at these reports to see if the data has been properly identified as sensitive. Some reports are written in concise form and must be formatted using the REPORT function. After running the REPORT function, you may want to give feedback to Data Privacy for Diagnostics Analyzer regarding some of the data that it either found as sensitive but was not actually sensitive, or feedback on data that was sensitive but not detected as sensitive. The FEEDBACK function allows you to perform this task. The cycle of ANALYZE / REPORT / FEEDBACK provides a way to train the Data Privacy for Diagnostics Analyzer processing in order to produce dumps with the right level of redaction for your environment.

Another function that can be used is the INGEST function. This allows you to import data from databases and files, and lets you create custom information that can be used by the Data Privacy for Diagnostics Analyzer processing to help identify sensitive data.

In order to display the exact criteria that the ANALYZE function is using to determine data sensitivity, one could use the EXTRACT function to write out any built-in or custom identifiers to a file such that when that particular identifier is requested in the ANALYZE configuration, the user knows exactly which tokens or what pattern will be used to mark data as sensitive or non-sensitive.

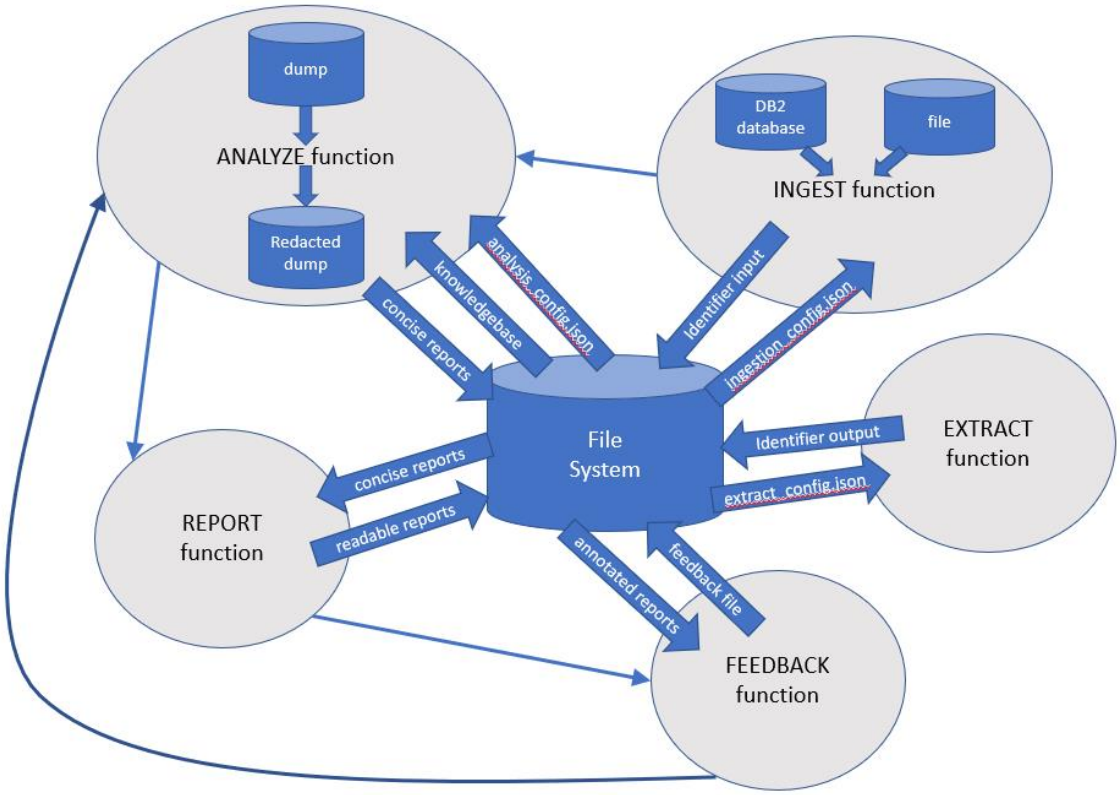


Figure 1. Data Privacy for Diagnostics Usage Cycle

*Using the Data Privacy for Diagnostics* Analyzer *Dialog within IPCS*
When IPCS is used, panels are presented to allow you to specify parameters required for processing. The dialog generates appropriate JCL based on the parameters provided. If any datasets are required but not pre-allocated, the dialog attempts to dynamically allocate them. If dynamic allocation fails for any reason, you should be able to pre-allocate datasets using other mechanisms (such as ISPF option 3.2).

The parameters specified on the IPCS Data Privacy for Diagnostics Analyzer panels are:
NOTE: not all parameters are present on all IPCS panels for each function

- INPUT, OUTPUT and TEMPORARY DATA SET NAME
    - Input and output data set names are required. Temporary data set names can either be a specific name or a data set name pattern. See the help panels for more information on patterns.
- BYPASS DP ANALYSIS

    - Allows you to submit a job that will either perform analysis (N) or skip analysis (Y). If N is specified, the Data Privacy for Diagnostics Analyzer step will scan the input data set looking for additional sensitive data in addition to data identified by the applications that allocated the storage marked as sensitive. If found, either token-level or page level redaction will be performed based on the Allow Page Level specification. If Y is specified, this step will be bypassed. The output data set identified by the NEW DATA SET NAME field will only have data removed that was identified by the applications that allocated the storage marked as sensitive.
- REDACTION STRING
    - If you are not allowing page level redaction, this redaction string is used to overlay data determined to be sensitive in the output dump. You may leave this field blank to overlay the token with X or specify a string. When longer strings are detected in the pages, the string is used in a repeated fashion. If shorter strings are found, only a portion of the redaction string may be used.
- NUMBER OF THREADS
    - For ANALYZE requests, large dumps may be processed faster by using multi-threading. You may specify 1 to 8 for the number of threads. Each thread requested will process a portion of the input dump, reducing the elapsed time it takes to process the entire dump, however, it may also increase the simultaneous amount of resources required to process the request. NOTE: Requesting additional threads and/or including additional identifiers will increase the size of the heap for the JVM, so use the –Xms and –Xmx options to adjust the minimum and maximum heap size.
- ALLOW PAGE LEVEL
    - If Y is specified, known as fast-analysis mode, the entire page of storage is redacted when any sensitive data is detected. If N is specified, known as detailed analysis mode, only the strings that are determined to be sensitive are overlaid with the redaction string. Specifying Y may allow the analysis processing to run faster since processing will stop at the first sensitive string in a page is found. However, it is possible that allowing page level of data omission may cause the integrity of the diagnostic data to be compromised. If you find this to be true, set the value to N so that data must be redacted using only the redaction string.
- SENSITIVE REPORT
    - If Y is specified, human-readable reports are generated in <directory>/reports/<dump-name>/<run-

number>/sensitive_token_log_*n*. There will be a file per thread requested. For each string detected, data is written to these files to help you understand what has been redacted and why. Based on this information, you may decide to include or exclude types of data.

- DPfD HOME DIR
    - Specify the path where the Data Privacy for Diagnostics Analyzer home directory is configured, <directory> as previously described. Do NOT include the trailing / when specifying this path.
- JAVA HOME DIR
    - The dialog will use the "whence java" command to try to determine where java is installed. If this can be determined, the path will be shown. This will be used in the batch job's STDENV set up file to create the proper environment for the java processing to run in. Do NOT include the trailing / when specifying this path.
- JAVA OPTIONS
    - You may provide whatever java options are desired. For example, you may need to specify a minimum and maximum heap size for the JVM to successfully run a multi-threaded ANALYZE request. Using the default setup with only built-in identifiers, each thread requires approximately 512MB to successfully load data for the run. Requesting additional threads and/or including additional identifiers will increase the size of the heap for the JVM, so use the –Xms and –Xmx options to adjust the minimum and maximum heap size. For more information on JVM Command-Line Options, please see the topic OpenJ9 command-line options in IBM SDK, Java Technology Edition 8.0.0 (https://www.ibm.com/support/knowledgecenter/en/SSYKE2_8.0.0/openj9/cmdline_specifying/index.html)
- JZOS LOAD MODULE
    - The dialog uses the JZOS Batch Launcher in the JCL that is submitted. You should determine the correct level of JZOS installed on your system and provide the name of the appropriate load module in this parameter. For example, the 64-bit version 8 load module for JZOS Batch Launcher is JVMLDM86. For additional information, please see the JZOS Batch Launcher and Toolkit Installation and Users Guide. (https://www.ibm.com/support/knowledgecenter/en/SSYKE2_8.0.0/com.ibm.java.zsecurity.80.doc/zsecurity-component/jzos.html)
- MIGLIB DATASET
    - A sort E35 exit is used to remove pages flagged as sensitive. This function is provided in module BLSRTE35 which is shipped in SYS1.MIGLIB. Should you need to override where this exit can be

loaded from, provide the name of the MIGLIB that contains the load module you wish to run.

- TEMP ALLOC PARMS
    - If your environment requires specific allocation parameters for dump datasets, you may supply any allocation parameters that will ensure the data set is properly allocated. For example, supplying DATACLAS and STORCLAS keywords may be necessary to locate the correct storage pool and attributes.
    - NOTE: Do NOT specify RECFM, DSORG, LRECL, BLKSIZE, SPACE and TRACK as they are used to create some of the interim data sets. If you need to use one of those allocation parameters, please request the ANALYZE function via the JCL instead of through IPCS.
- EDIT CONFIG FILE?
    - If Y, allows the user to edit the configuration file pertaining to the function requested (analysis_config.json for ANALYZE, ~~or~~ ingestion_config.json for INGEST or extract_config.json for EXTRACT) prior to submitting the JCL to perform the requested function. Default is N. Please see the analysis_config.json, extract_config.json and ingestion_config.json sections for additional information.
- RUN NUMBER
    - From the ANALYZE step, a run number was generated and can be found in the job output which can be specified for this parameter when the function requested is REPORT or FEEDBACK. If a run number is NOT specified, the most recent ANALYZE run for the input dump is used.
- DB2 JDBC PATH
    - For the INGEST function, if using database as the source in the ingestion_config.json file, this field is needed to specify the path for the DB2 JDBC Driver and License JAR files. Do NOT include the trailing / when specifying this path.

## Requesting an ANALYZE run

ANALYZE is the function that will look for sensitive data in dump records and flag those records for redaction, or even overlay that sensitive data with a redaction string if token-level redaction is requested. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPA JCL in SYS1.SAMPLIB) , two important configuration files are used.

- Runtime configuration file

    - This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPA JCL.
- analysis_config.json file
    - This file provides additional detail on what to include or exclude while looking for sensitive data in dump records. It is located in the <directory>/configuration/ directory in the file system. You may either use the EDIT CONFIG FILE option Y in the ANALYZE IPCS panel to edit this file if you want to change it, or you may directly edit it using an editor that you are familiar with.

The Runtime configuration is a json file that is built by the dialog using the parameters supplied on the panel, or can also be supplied as a file or an instream dataset in JCL if you use JCL to submit the job. The runtime configuration file parameters (hand-coded in the BLSJDPA JCL) are:

- "thread_count" : This specifies the number of worker threads to be spawned. The total number of threads is one more than this (there is one monitor thread). If omitted, the default value is 4 for the JCL interface.
- "input_dataset" : Specifies the location of the input dump. You must specify a dataset name as follows: "//'<dump-dataset-name>'"
- "record_count" : Estimated number of records in the input dump. If set to 0 or omitted, the Data Privacy for Diagnostics Analyzer will count the actual number of records. This is used when multiple threads are requested to ensure that the records are evenly split by the requested number of threads. This parameter is not available in IPCS panel interface.
- "output_dataset_prefix" or "output_dataset" : Specifies either the prefix to use for each thread's output dump data, or the list of dataset(s). Either dataset names or DD names may be specified on this parameter. For example, "output_dataset_prefix":"//'SYS1.DUMP.D190926.T132348'" indicates the prefix that will be used by each thread. Files will either be dynamically allocated or may be pre-allocated as SYS1.DUMP.D190926.T132348.F1, SYS1.DUMP.D190926.T132348.F2, SYS1.DUMP.D190926.T132348.F3, etc., one per thread. When using the BLSJDPA JCL, you may also specify DDNAMEs as prefixes. For example, you may specify "output_dataset_prefix":"//DD:ANLZO" and specify DD statements for //ANLZOF1, //ANLZOF2, etc, for each thread's output. Alternatively, you may use the "output_dataset" method of supplying a list of datasets. For example, you may specify "output_dataset":["//'SYS1.DUMP.D190926.T132348.F1'","//'SYS1.DUMP.D190926.T132348.F2'"]. If you are using the dialog to initiate the job, you may also specify a pattern to be used. In this case, the pattern may contain a

single "%" character which will cause the dialog to generate dataset names with thread numbers substituted in that position in the dataset name. For example, you may specify 'SYS1.DUMP.D190926.T132348.F%' as the dataset name pattern on the panel and the dialog would generate "output_dataset":["//'SYS1.DUMP.D190926.T132348.F1'","//'SYS1.DUMP.D190926.T132348.F2'"] as the parameters.

- "dpfd_home" : Specifies the Data Privacy for Diagnostics Analyzer home directory
- "redaction_string" : String to use for redaction for pages being analyzed using detailed analysis. When a redaction_string isn't specified and detailed analysis is specified, sensitive data will be replaced with X.
- "analysis_mode" : Specifies the analysis mode for detecting sensitive data. Valid values are 1 for Fast Analysis and 2 for Detailed Analysis. In Fast mode, the entire page is marked as sensitive as soon as first sensitive token is identified in the page. In Detailed mode, each sensitive token is identified independently and overlaid with the redaction_string. Note that when 1 is specified, some pages may be analyzed using detailed analysis mode.
- "character_set" :Specified the character set that should be used for decoding of the input dump. The default value is "Cp1047". This parameter is not available in IPCS panel interface.
- "log_sensitive_tokens" : Specifies if the sensitive token log for each file should be generated. When set to TRUE, a sensitive token log for each thread should be generated in the <directory>/reports folder.

The analysis_config.json file:
As your experience with this ANALYZE processing matures, this file will likely become stable until major changes in data occur in your environment. You will likely start with the default file supplied with the product. As your usage evolves, you may decide to exclude or include certain built-in identifiers, use custom identifiers or add dependent identifiers.

*Parameter Descriptions:*

- "built_in_identifiers_include" : This specifies the built-in identifiers which should be used to detect sensitive tokens. Values that can be supplied here are listed later in this documentation <insert reference>. If nothing is specified, no identifiers are included.
- "built_in_identifiers_exclude" : This specifies the built-in identifiers which should not be used to detect type of tokens. Values that can be supplied here

are listed later in this documentation <insert reference>. If nothing is specified, no identifiers are excluded.

- "custom_identifiers" : This specifies any custom identifiers which should be used to detect sensitive data. Custom identifiers can be ingested using the INGEST function and can be in the form of dictionaries or patterns. Arrays of multiple identifiers may be specified. Each identifier has the following fields:
    - "format" : Valid value is "custom". "custom" indicates that a previously ingested dictionary or set of patterns is specified.
    - "inputfilename" : File name containing the identifier data. If the format is "custom", the Data Privacy for Diagnostics Analyzer looks for the file in /<directory>/knowledgebase/ingested/.
    - "entitytype" : Specifies a name for the identifier. This is used when it is part of dependent identifier. If the entity name was provided when the file was ingested (for "custom" format), this field can be skipped. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.
    - "description" : Specifies description of the identifier. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.

- "dependent_identifiers" : This specifies a set of identifiers which are sensitive only when they all occur in a page. Here, you specify an array where multiple dependent identifiers can be specified. When an identifier is made part of dependent_identifier, it stops being an independent identifier. Note that all of the identifiers specified in a dependent_identifier should be present in built_in_identifiers_include or in custom_identifiers; otherwise, this will never be detected. Each dependent identifier has the following fields:
    - "name" : Specifies name assigned to this dependent identifier.
    - "identifiers" : Specifies the set of identifier belonging to this dependent identifier.
- “built_in_ns_identifiers_include” : This specifies the built-in identifiers which should be used to detect that a token is non-sensitive.
- “built_in_ns_identifiers_exclude” : This specifies the built-in identifiers which should not be used to detect that a token is non-sensitive.
- “custom_ns_identifiers” : This specifies any custom identifiers which should be used to identify tokens as non-sensitive data. Custom identifiers can be ingested using the INGEST function and can be in the form of dictionaries or

patterns. Arrays of multiple identifiers may be specified. Each identifier has the following fields:
    - "format" : Valid value is "custom". "custom" indicates that a previously ingested dictionary or set of patterns is specified.
    - "inputfilename" : File name containing the identifier data. If the format is "custom", the Data Privacy for Diagnostics Analyzer looks for the file in /<directory>/knowledgebase/ingested/.
    - "entitytype" : Specifies a name for the identifier. This is used when it is part of dependent identifier. If the entity name was provided when the file was ingested (for "custom" format), then this field can be skipped. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.
    - "description" : Specifies description of the identifier. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.
- “printable_characters”: This field specifies set of printable characters which will be used for analysis of dumps. When analyzing the dump, only these characters are used to construct the tokens that will be parsed. Default value is "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890` ~!@#$%^&*()-=_+ {}[];:'\",<.>/?\\|\n "

Any incorrect information in analysis_config.json will ignored.

Identifiers specified in the built_in_identifiers_exclude list take precedence over identifiers specified in the built_in_identifiers_include list.

*analysis_config.json example:*

```
{
    "built_in_identifiers_include" : ["Month", "FullName", "Credit Card Type", "Credit Card Number", "Email", "Day"],
    "built_in_identifiers_exclude": ["Year", "Zipcode", "Date Time"],
    "custom_identifiers": [
        {
            "inputfilename" : "acctnum.bin",
            "entitytype" : "Account Number",
            "description" : "List of account numbers",
```

```
                "format" : "custom"
            },
             {
                "inputfilename" : "policynum.bin",
                "entitytype" : "PolicyNumber",
                "description" : "List of policy numbers",
                "format" : "custom"
            }
          ],
        "dependent_identifiers": [
            {
                "name": "Full Person",
                "identifiers": [ "FullName", "Zipcode", "Email" ]
            },
            {
                "name": "Card",
                "identifiers": [ "Credit Card Type", "Credit Card Number"]
            }
        ]
    }
```

The built_in_identifiers supplied with Data Privacy for Diagnostics Analyzer are (by their very nature are not necessarily all inclusive of the particular topic, and may not be changed over time should there be changes to the actual data set of said topic) :

| Identifier (not case sensitive) | Description |
|---|---|
| age | String patterns related to age.<br>Examples:<br>“10 years old”<br>“4 months old”<br>“dob: 1-2-1999"<br>NOTE: These string patterns will be considered sensitive, but just the number 10 (in the first example) will not be considered sensitive by itself. |
| animal | Identifies animals using a species dictionary. |
| atc | Dictionary containing values for the Anatomical Therapeutic Chemical (ATC) classification system. |

| | |
|---|---|
| continent | Dictionary containing the names of continents. |
| country | Dictionary containing the names of countries. |
| county | Dictionary containing the names of all the counties in US. In the United States of America, an administrative or political subdivision of a state is a county. |
| credit card type | Credit card identification dictionary. Cards detected are VISA, Mastercard, AMEX, Diners Club, Discover and JCB |
| credit card number | Credit card pattern identification. Card patterns detected are VISA, Mastercard, AMEX, Diners Club, Discover and JCB |
| date time | Date and Time pattern identification |
| day | Dictionary containing the names of the days of the week |
| dependent | Dictionary containing the names of types of dependents, such as daughter, son, etc. |
| email | Email addresses pattern |
| eu nin | National Identification Number patterns for various EU countries |
| FullName | A first name and last name pair dictionary, the combination of which is from popular names in the US census |
| gender | Dictionary containing the genders Male and Female |
| hospital name | Dictionary containing the names of hospitals in the US. |
| iban | International Bank Account Number (IBAN) pattern |
| icdv9 | International Classification of Diseases 9th Revision (ICDv9) identification dictionary |
| icdv10 | International Classification of Diseases 10th Revision (ICDv10) identification dictionary |

| | |
|---|---|
| imei | International Mobile Equipment Identity (IMEI) identification dictionary. |
| imsi | International Mobile Subscriber Identity (IMSI) Identification dictionary |
| in aadhaar card number | Aadhaar identification number pattern for residents or passport holders of India |
| in pan card number | Permanent Account Number pattern issued by Indian Income Tax Department |
| international phone number | International phone number identification pattern |
| ip address | IP address identification pattern. Supports both IPv4 and IPv6 addresses |
| latitude longitude | Latitude/longitude identification pattern. Supports GPS and DMS coordinates formats, Ex: 12:30'23.256547S 12:30'23.256547E, N90.00.00 E180.00.00 |
| mac address | MAC Address Identification pattern |
| marital status | Marital status identifier dictionary |
| medical name | Medical Name identification pattern. Example: John Doe MD |
| medical record number | Medical Record Number identification pattern, for example, MRN: CLM-00000056055, Medical Record Number: 1234asds |
| month | Month Name identification dictionary |
| occupation | Occupation identification dictionary |
| phone number | Phone number identification pattern. |
| PO box | Pattern that identifies post office box numbers with PO Box prefix. |
| raceOrEthnicity | Dictionary identification of ethnic groups |
| religion | Dictionary containing major religions |
| street types | Street Type identification dictionary, for example, tokens containing "st." |
| uk nin | National Insurance Number pattern |
| url | URL identification pattern. Supports HTTP and HTTPS detection |

| | |
|---|---|
| us address | Identifies US-centric address patterns like “800 Theatre Court Garden City, NY 11530”. This just checks the format but does not validate city and state/zip in the address. |
| us phone number | US specific phone/fax/pager identifier pattern |
| us ssn | US Social Security Number pattern |
| us states | US State Name identification dictionary |
| us swift code | US SWIFT Code identification pattern, Ex: AIBKUS3TTMK |
| vehicle identification number | Vehicle identification number identification dictionary |
| year | Year of Birth Identification pattern, Any number between 0 and current year. |
| zipcode | Valid US zip code identifier dictionary. |

NOTE: For additional details on built-in sensitive identifier dictionaries or patterns, use the EXTRACT function.

The list of built_in_ns_identifiers supplied with Data Privacy for Diagnostics Analyzer are:

| Identifier (not case sensitive) | Description |
|---|---|
| moduleName | This is used to avoid classifying programs (CSECTs within load modules) as sensitive data. The module name identifier works as follows :-This identifier detects the patterns of following types : <module name> <date time> <fmid>. The date time can be either 5 or 7 or 8 characters long. Module name can be up to 8 characters long. Fmid can be either 7 or 8 characters long. Module name and date time are mandatory. Fmid is optional. Data Privacy for Diagnostics has a built-in set of module name prefixes and fmid prefixes that IBM supplies. The module names prefixes can be increased by adding them to |

| | |
|---|---|
| | /<directory>/configuration/moduleprefix.txt and the fmid prefixes can be increased by adding them to /<directory>/configuration/fmid.txt |

The default analysis_config.json file will include the list of included identifiers and excluded identifiers.

Identifier checking is done using the following order:

1. User feedback indicates that a token is sensitive
2. User feedback indicates that a token is non-sensitive
3. module name check
4. Non sensitive checks (built-in + custom)
5. sensitive checks (built-in + custom)

## Requesting a REPORT run

REPORT is the function that will format concise reports in the file system into human-readable reports for the requested dump, and serves as the input to the FEEDBACK function. By default, the reports for the last ANALYZE run will be formatted. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPR JCL), the runtime configuration file is used to specify options for processing. This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPR JCL. The runtime configuration file parameters (built by dialog or hand-coded in the BLSJDPR JCL) are:

- "input_dataset" : Specifies the location of the input dump. You must specify a dataset name as follows: "//'<dump-dataset-name>'". During ANALYZE processing, the dump name was used as a directory and will be used by REPORT processing to locate the files produced during ANALYZE processing.
- "dpfd_home" : Specifies the Data Privacy for Diagnostics Analyzer home directory
- "run_number" : Specifies the run number. If you omit the run_number option, it will use the most recent ANALYZE run.

As you validate how accurate the ANALYZE processing was in detecting sensitive data in your dumps, you may need to provide feedback to help the ANALYZE processing. In order to provide this feedback, you need to run the REPORT processing prior to the FEEDBACK function. The outputs of the report processing are human-readable files that can be edited should you need to provide FEEDBACK for future ANALYZE attempts. The files produced from the REPORT function are:

- /<directory>/reports/<dump-name>/<run-number>/sensitive_tokens : this file contains the list of each token that was found to be sensitive data.
- /<directory>/reports/<dump-name>/<run-number>/non_sensitive_tokens : this file, if requested, contains the list of each token that was found to be non-sensitive data. Note that this file is only produced if ALLOW PAGE LEVEL is "N" during the ANALYZE processing.

## Requesting a FEEDBACK run

After running the ANALYZE processing, followed by the REPORT processing, you may want to provide feedback to enhance the accuracy of future ANALYZE functions in detecting the appropriate sensitive data for your environment. You can provide feedback to indicate the following:

- Tokens found to be sensitive are not actually sensitive
- Tokens not found to be sensitive are actually sensitive

To do this, you must edit the reports generated from the REPORT processing. These reports will be found in the /<directory>/reports/<dump-name>/<run-number> directory.

- In the sensitive_tokens file, change the "Is_Analysis_Correct" field from "Y" to "N" for any token that should not be considered sensitive.
- In the non_sensitive_tokens file, change the "Is_Analysis_Correct" field from "Y" to "N" for any token that should be considered sensitive.

Afterwards, the FEEDBACK function can be requested. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPF JCL), the runtime configuration file is used to specify options for processing. This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPF JCL. The runtime configuration file parameters (built by dialog or hand-coded in the BLSJDPF JCL) are:

- "input_dataset" : Specifies the location of the input dump. You must specify a dataset name as follows: "//'<dump-dataset-name>'". During ANALYZE processing, the dump name was used as a directory and will be used by FEEDBACK processing to locate the files produced during ANALYZE processing.
- "dpfd_home" : Specifies the Data Privacy for Diagnostics Analyzer home directory

- "run_number" : Specifies the run number. If you omit the run_number option, it will use the most recent ANALYZE run. Ensure that the run_number is the same directory in which the edited reports are contained.

During the FEEDBACK operation, the Data Privacy for Diagnostics Analyzer reads these edited reports and updates the <directory>/knowledgebase/feedback/feedback.bin file, which will be used for future ANALYZE runs.

## Requesting an INGEST run

You may want to customize the detection of sensitive data that is unique to your environment. This can be achieved by the INGEST function, which will help the Data Privacy for Diagnostics Analyzer to detect the sensitive data in future analysis. You can initiate the INGEST function from either IPCS option 5.6 or the BLSJDPI JCL. Either way, it will use the following files:

- Runtime configuration file
    - This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPI JCL.
- ingestion_config.json file
    - This file provides detail on what additional data is to be used by the Analyzer while analyzing diagnostic data. It is located in the <directory>/configuration/ directory in the Data Privacy for Diagnostics file system. You may either use the EDIT CONFIG FILE option Y in the INGEST IPCS panel to edit this file if you want to change it, or you may directly edit it using an editor that you are familiar with.

Regardless of how the job is initiated, the runtime configuration file is used to specify options for processing. The runtime configuration file parameters (Built by dialog or hand-coded in the BLSJDPI JCL) are:

- " dpfd_home" : Specifies the Data Privacy for Diagnostics Analyzer home directory

The ingestion_config.json file: This file contains information about the identifiers to be built as sensitive or non-sensitive tokens based on the options specified. INGEST will generate a file in the <directory>/knowledgebase/ingested directory, which can be subsequently specified in the analysis_config.json file to add the ingested identifier as a custom identifier for sensitive data detection during the ANALYZE function. This data can be ingested from dictionaries, databases or other sources.

*Parameter Descriptions:*

- “outputfilename”: Specifies the name of the file to be stored in the <directory>/knowledgebase/ingested directory after successful INGEST run.

This can be specified in the analysis_config.json file as the inputfilename under the custom_identifiers option for use in future ANALYZE requests.

- “entitytype”: Specifies the name of the identifier. This is used when user does not provide one in the analysis_config.json file in the custom_identifiers option under entitytype.
- “description”: Specifies a description of the identifier. This is used in future REPORTs when user does not provide one in the analysis_config.json file in the custom_identifiers option under description.
- "inputtype" : Specifies the type of input to INGEST. Valid values are:
    - “pattern” Allows the specification of a Java Regex (or Java Regular Expression) as a pattern for matching strings.
    - “dictionary” Allows specification of exact tokens to be matched by the Data Privacy for Diagnostics Analyzer.
- “inputsource”: Specifies the source of the input data to be ingested. Valid values are:
    - “file” Indicates that the source is a file with the location specified on the “inputfilename” option.
    - “inline” Indicates that the source is inline data specified in the ingestion_config.json file.
    - “database” Indicates that the source is a database as specified in the “database” and associated options. This option is only valid when “inputtype” of ”dictionary” is specified.
      NOTE: If you chose this option, you MUST update the DB2 JDBC PATH, by either entering the full path to the DB2 JDBC Driver and License JARs in the DB2 JDBC PATH field in IPCS option 5.6 or update the CLASSPATH section of the STDENV DD in the BLSJDPI JCL to include the full path.
- “inputfilename”: Specifies the path and name of the file which contains the data to be used during the INGEST function. This option is only valid when “inputsource”:”file” is specified. The format for specifying a dictionary or pattern in a file is 1 entry per line as such:
  value1
  value2
  value3
- “inlinedata”: Specifies the data to be used during the INGEST function. This option is only valid when “inputsource”:”inline” is specified. The format for specifying inline data is: "inlinedata" : ["value1", "value2", "value3"]
- “database”: Specifies the type of database to be used as an input source. This option is only valid when “inputsource” of ”database” is specified. Valid values are:

    - “DB2" Indicates the database is DB2 installed on non-system Z platform
    - “DB2zOS” Indicates the database is DB2 installed on system-Z
    - “DB2zOSptkt” Indicates the database is DB2 to be connected via pass ticket.
        - This is the default value.
- “databasehost”: Specifies the domain name or IP address where the database is hosted. This option is only valid when “inputsource” of ”database” is specified. The format for specifying this option is: “databasehost”:”<url>”
- “databaseport”: Specifies the port number that identifies the DB2 subsystem. This option is only valid when “inputsource”:”database” is specified.
- “databaseusername”: Specifies the user ID used to connect to the DB2 database. This option is only valid when “inputsource”:”database” is specified.
- “databasepassword”: Specifies the password for the user ID used to connect to the DB2 database. This option is only valid when “inputsource”:”database” is specified.
- “databasename”: Specifies the name of the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
- “databaseschema”: Specifies the schema in the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
- “databasetablename”: Specifies the name of the table in the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
- “databasecolumnname”: Specifies the name of the column in the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.

Any options specified in the ingestion_config.json file which are not valid with the specified inputsource will be ignored. Only 1 value per option will be processed, except for the inlinedata option which may be specified in a list form.
After the INGEST request is completed, the newly created identifier must be specified in the custom_identifiers options of the analysis_config.json file in order to be considered for determining sensitive data for the subsequent ANALYZE requests.

ingestion_config.json examples:

If you want to create a new sensitive identifier called account that will detect the account number of your customers in a dump, you can use a pattern if you know that all of the account numbers take a certain format such as 2 alphabetic characters followed by 8 numeric digits. The following is an inline pattern using Java Regex:

```
{
"inputtype":"pattern",
"inputsource":"inline",
"entitytype":"account",
"description":"a pattern to determine account: 2 characters, 8 digits",
"outputfilename":"accts.bin",
"inlinedata":["\\D{2}\\d{8}"]
}
```

Note that \ is an escape character in Java strings, which requires you to use \\ to define a single \ in order to use meta Regex characters like \D (non-digit) and \d (digit).

If you have a file that contains all of the account numbers for your customers, you can alternatively provide a file dictionary to create your custom identifier:

```
{
"inputtype":"dictionary",
"inputsource":"file",
"entitytype":"accounts",
"description":"a list of our customer accounts",
"outputfilename":"accts.bin",
"inputfilename":"/u/ibmuser/accounts.txt"
}
```

You may also INGEST a column from a DB2 database as a dictionary that can be used to identify sensitive data:

```
{
"inputtype":"dictionary",
"inputsource":"database",
"entitytype":"accounts",
"description":"a list of our customer accounts",
"outputfilename":"accts.bin",
"database":"db2zos",
"databasehost":"db2host.pok.ibm.com",
"databaseport":"446",
"databaseusername":"db2user",
"databasepassword":"Bot27tle",
```

```
"databaselocation":"DBX5LOC1",
"databaseschema":"dsn8910",
"databasetablename":"DBO",
"databasecolumnname":"ACCOUNTNUMBER"
}
```

After the INGEST process has been completed, in order to use your newly created customer identifier in the subsequent ANALYZE request, you must amend your analysis_config.json file to add the accounts to the custom identifier as such:

```
"custom_identifiers":
[
        {
         "inputfilename" : "accts.bin",
         "entitytype" : "accounts",
         "description" : "Customer Accounts",
         "format" : "custom"
         }
]
```

### Requesting an EXTRACT run

You may want to know what criteria that ANALYZE is using to determine the sensitivity of data.
This can be achieved by the EXTRACT function, which will write the pattern or dictionary of an identifier (built-in or custom) to a file. You can initiate the EXTRACT function from either IPCS option 5.6 or the BLSJDPX JCL. Either way, it will use the following files:

- Runtime configuration file
    - This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPX JCL.
- extract_config.json file
    - This file provides additional detail on what to include or exclude while looking for sensitive data in dump records. It is located in the <home>/configuration/ directory in the Data Privacy for Diagnostics file system. You may either use the EDIT CONFIG FILE option Y in the EXTRACT IPCS panel to edit this file if you want to change it, or you may directly edit it using an editor that you are familiar with.

Regardless of how the job is initiated, the runtime configuration file is used to

specify options for processing.  The runtime configuration file parameters (Built by dialog or hand-coded in the BLSJDPX JCL) are:

- " dpfd_home" : Specifies the Data Privacy for Diagnostics Analyzer home directory

The extract_config.json file: This file contains information about the identifiers to be written to a file based on the options specified. EXTRACT will generate a file in the specified output directory. The identifiers can be either built-in or custom from a prior INGEST operation.

*Parameter Descriptions:*

- “identifiers_list”: Specifies the array of identifers to be written to a file. Each identifier should have a different output file as to not over-write the previous identifier. The format for specifying an identifiers_list is:
  “identifiers_list”:[
  {“identifier_type”:”<type1>”,”identifier_name”:”<name1>”,”output_filename”:”<outputpath1>”},
  {“identifier_type”:”<type2>”,”identifier_name”:”<name2>”,”output_filename”:”<outputpath2>”}]
  - “identifier_type”: Specifies the type of the identifier. Valid values are “builtin” for built-in identifier or “custom” for a custom identifier from an INGEST request.
  - “identifier_name”: Specifies the name of the identifier.
    - If a custom identifier is specified, this is file name in the knowledgebase/ingested directory.
    - If a built-in identifier is specified, this is the name of the built-in identifier as defined in *Chart On Page 11 in ANALYZE*
  - “output_filename”: Specifies the full path for the file where the extracted information will be written.

extract_config.json example:

If you want to extract both a built-in identifier (person) and custom identifier (the accounts identifier that we have previously defined in the INGEST request section), then your identifiers_list would be the following:

```
{
“identifiers_list”:
[
        {
        “identifier_type”:”builtin”,
```

```
        ”identifier_name”:”FullName”,
        ”output_filename”:”/DPfD/names.txt”
        },
        {
        “identifier_type”:”custom”,
        ”identifier_name”:”accts.bin”,
        ”output_filename”:” /DPfD/accounts.txt”
        }
]
}
```

# z/OS MVS IPCS Commands (SA23-1382-00)

## Topic: IPCS CLISTs and REXX EXECs

[Add line & Subtopic]

BLSXREDR REXX EXEC — Report pages marked as sensitive

Use the BLSXREDR EXEC to generate a report about the parameters used to perform analysis via Data Privacy for Diagnostics to produce the redacted dump (if the ANALYZE function was used via the IPCS panels, but not via the BLSJDPA JCL) as well as ranges of pages in a dump which were tagged with the SENSITIVE=YES attribute (if the dump is not post-processed) or redacted by the Data Privacy for Diagnostics Analyzer and being tagged with the SENSITIVE=YES attribute (if the dump is post-processed).

- Syntax
  - %BLSXREDR <dump_dsn> [A <asid>] [DETAILS]
- Parameters
  - <dump_dsn>
    - Specify the name of a post-processed dump data set.
    - NOTE: If using BLSXREDR within IPCS, this parameter cannot be specified, and the currently ACTIVE dump data set will be used.
  - A <asid>
    - [Optional] Specify an address space identifier number in hex as a filter to reduce the output to contain only addresses for a specific address space. If omitted, all ASIDs will be displayed.
    - NOTE: ASID can be used instead of A.
  - DETAILS
    - [Optional] When specified, displays all pages that were tagged as SENSITIVE=NO in addition to pages which were partially redacted by the Data Privacy for Diagnostics Analyzer (for post-processed dumps) or marked redactable via being tagged SENSITIVE=YES (for not post-processed dumps). This does NOT display page ranges in which entire pages were removed from the post-processed dump due to being tagged as SENSITIVE=YES or via the Data Privacy for Diagnostics Analyzer when Allow Page Level was set to Y during the ANALYZE function for post-processed dumps.
    - NOTE: If using BLSXREDR within IPCS, this parameter cannot be specified.

- JCL invocation

This JCL will run the BLSXREDR EXEC against the specified post-processed dump data set and display information about tagged and redacted pages for all address spaces:

```
//XREDR    EXEC PGM=IKJEFT01
//SYSEXEC  DD DISP=SHR,DSN=SYS1.SBLSCLI0
//SYSTSPRT DD SYSOUT=A
//SYSTSIN  DD *
  %BLSXREDR 'SY1.DUMP.D200130.SY1.S00003.REDACTED'
/*
```

- Operator console invocation

This command will run the BLSXREDR EXEC against the specified post-processed dump data set and use the optional DETAILS parameter to display information about pages in all address spaces tagged the SENSITIVE=NO and any pages partially redacted by the Data Privacy for Diagnostics Analyzer:

```
%BLSXREDR 'SY1.DUMP.D200130.SY1.S00003.REDACTED' DETAILS
```

- IPCS dialog invocation

BLSXREDR produces a report of the addresses of pages in the specified address space that were marked as containing sensitive data in the current active redacted dump data set:

```
---------------------------- IPCS Subcommand Entry ------------------------
Enter a free-form IPCS subcommand, CLIST, or REXX EXEC invocation below:
===] %BLSXREDR ASID 001F
```