Cribl

>DEPLOYMENT GUIDE_

Zscaler NSS Integration

>DEPLOYMENT GUIDE_

Zscaler NSS Integration

Document Purpose:

This deployment guide shows how to set up Zscaler NSS to direct live network logs to Cribl Stream. From there, Stream can reduce and refine the data in flight for better handling and insights.

Zscaler Nanolog Streaming Service (NSS) uses a VM to stream traffic logs in real time from the Zscaler Nanolog to a SIEM. Cribl Stream can take the place of the SIEM in this arrangement. Then you can use Cribl Stream to greatly reduce the size of ZScaler logs.

Configuring NSS to Send Data to Cribl Stream

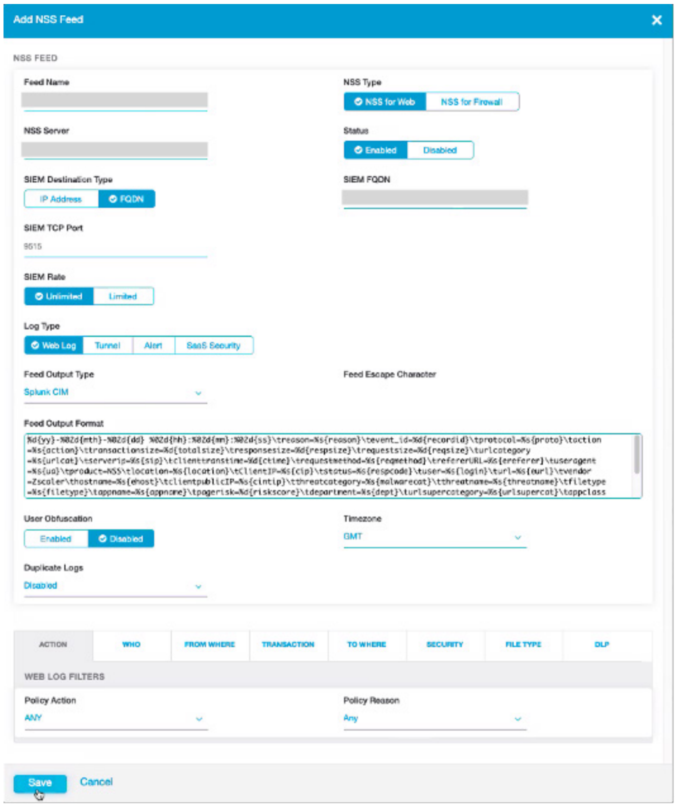
Since Nanolog forwards data to a single IP address or FQDN, Cribl recommends that you use a load balancer to distribute data among Cribl Stream Workers.

Nanolog delivers data using a raw TCP connection.

In Zscaler:

- Go to **Administration** > **Nanolog Streaming Service**.
- In the **NSS Feeds** tab, **click Add NSS Feed** to open the following configuration window:

Fig. 01:



Add NSS Feed

NSS FEED

Feed Name: [redacted]

NSS Type: ● NSS for Web ○ NSS for Firewall

NSS Server: [redacted]

Status: ● Enabled ○ Disabled

SIEM Destination Type: ○ IP Address ● FQDN

SIEM FQDN: [redacted]

SIEM TCP Port: 9515

SIEM Rate: ● Unlimited ○ Limited

Log Type: ● Web Log ○ Tunnel ○ Alert ○ SaaS Security

Feed Output Type: Splunk CIM

Feed Escape Character:

Feed Output Format:

```
%d{yy}-%02d{mth}-%02d{dd} %02d{hh}:%02d{mn}:%02d{ss}\treason=%s{reason}\tevent_id=%d{recordid}\tprotocol=%s{proto}\taction
=%s{action}\ttransactionsize=%d{totalsize}\tresponsesize=%d{respsize}\trequestsize=%d{reqsize}\turlcategory
=%s{urlcat}\tserverip=%s{sip}\tclienttranstime=%d{ctime}\trequestmethod=%s{reqmethod}\trefererURL=%s{ereferer}\tuseragent
=%s{ua}\tproduct=NSS\tlocation=%s{location}\tClientIP=%s{cip}\tstatus=%s{respcode}\tuser=%s{login}\turl=%s{eurl}\tvendor
=Zscaler\thostname=%s{ehost}\tclientpublicIP=%s{cintip}\tthreatcategory=%s{malwarecat}\tthreatname=%s{threatname}\tfiletype
=%s{filetype}\tappname=%s{appname}\tpagerisk=%d{riskscore}\tdepartment=%s{dept}\turlsupercategory=%s{urlsupercat}\tappclass
```

User Obfuscation: ○ Enabled ● Disabled

Timezone: GMT

Duplicate Logs: Disabled

WEB LOG FILTERS

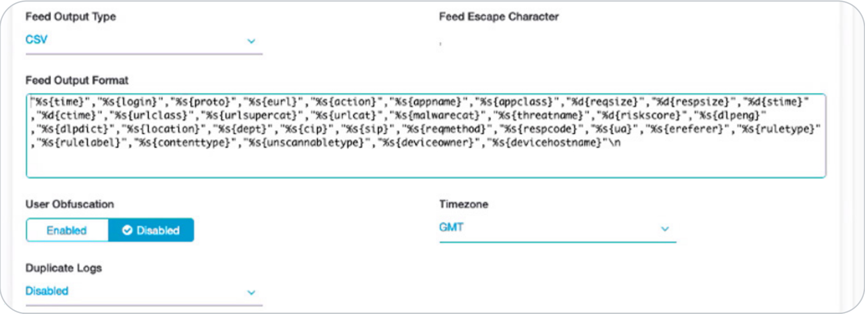
Policy Action: ANY

Policy Reason: Any

- Enter a Feed Name that identifies this feed as one that sends data to Cribl Stream.
- Enter the IP address or FQDN for either your Cribl Stream instance, or the load balancer you're using with your Cribl Stream instances.
- Select a **Feed Output Type**. `Splunk CIM`, a tab-delimited key/value format, is a typical choice.

Alternatively, you choose a different option, such as CSV:

Fig. 02:



Feed Output Type: CSV

Feed Escape Character: ,

Feed Output Format:

```
"%s{time}","%s{login}","%s{proto}","%s{eurl}","%s{action}","%s{appname}","%s{appclass}","%d{reqsize}","%d{respsize}","%d{stime}"
,"%d{ctime}","%s{urlclass}","%s{urlsupercat}","%s{urlcat}","%s{malwarecat}","%s{threatname}","%d{riskscore}","%s{dlpeng}"
,"%s{dlpdict}","%s{location}","%s{dept}","%s{cip}","%s{sip}","%s{reqmethod}","%s{respcode}","%s{ua}","%s{ereferer}","%s{ruletype}"
,"%s{rulelabel}","%s{contenttype}","%s{unscannabletype}","%s{deviceowner}","%s{devicehostname}"\n
```

User Obfuscation: ○ Enabled ● Disabled

Timezone: GMT

Duplicate Logs: Disabled

Example pipeline.

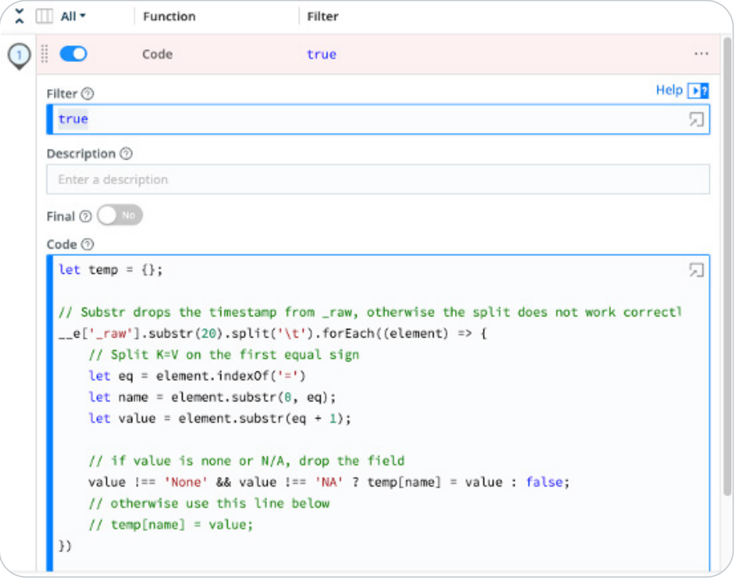
Cribl Stream can reduce Zscaler log size by (1) reformatting and reshaping the data, and (2) suppressing, sampling, and dropping appropriate fields.

The following code block shows how to correctly parse tab-delimited key-value pairs.

```
let temp = {};

// Substr drops the timestamp from _raw, otherwise the split does not work correctly
__e['_raw'].substr(20).split('\t').forEach((element) => {
    // Split K=V on the first equal sign
    let eq = element.indexOf('=')
    let name = element.substr(0, eq);
    let value = element.substr(eq + 1);

    // if value is none or N/A, drop the field
    value !== 'None' && value !== 'NA' ? temp[name] = value : false;
    // otherwise use this line below
    // temp[name] = value;
})

__e['_raw'] = temp;
```

Here's an example Pipeline that uses the parsing code above. (You can directly **import** this Pipeline in JSON form.)

A Code Function parses the data:



Filter: true

Description: Enter a description

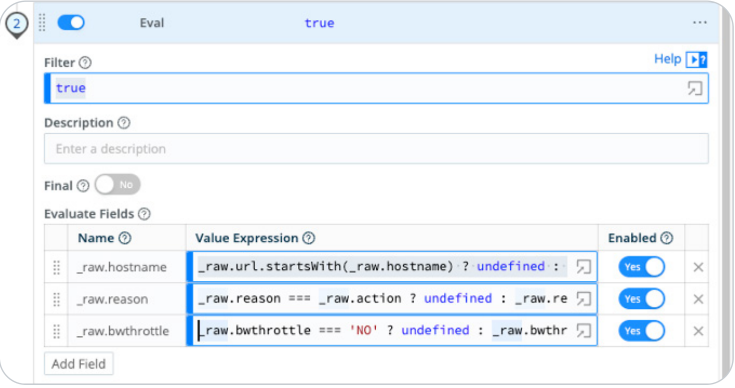
Final: No

Code:

```
let temp = {};

// Substr drops the timestamp from _raw, otherwise the split does not work correctl
__e['_raw'].substr(20).split('\t').forEach((element) => {
    // Split K=V on the first equal sign
    let eq = element.indexOf('=')
    let name = element.substr(0, eq);
    let value = element.substr(eq + 1);

    // if value is none or N/A, drop the field
    value !== 'None' && value !== 'NA' ? temp[name] = value : false;
    // otherwise use this line below
    // temp[name] = value;
})
```

Fig. 03:

An Eval Function reshapes the data:



Filter: true

Description: Enter a description

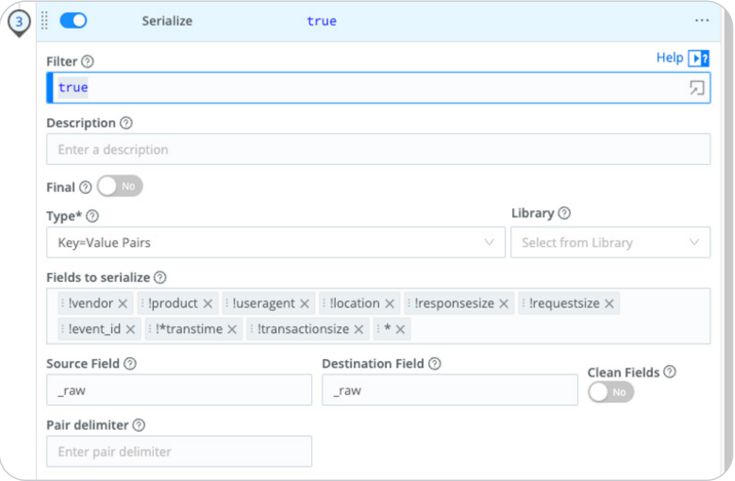
Final: No

Evaluate Fields

Name	Value Expression	Enabled
_raw.hostname	`_raw.url.startsWith(_raw.hostname) ? undefined :`	Yes
_raw.reason	`_raw.reason === _raw.action ? undefined : _raw.re`	Yes
_raw.bwthrottle	`_raw.bwthrottle === 'NO' ? undefined : _raw.bwthr`	Yes

Fig. 04:

And finally, a Serialize Function drops unwanted fields:



Filter: true

Description: Enter a description

Final: No

Type*: Key=Value Pairs

Library: Select from Library

Fields to serialize: !vendor, !product, !useragent, !location, !responsesize, !requestsize, !event_id, !*transtime, !transactionsize, *

Source Field: _raw

Destination Field: _raw

Clean Fields: No

Pair delimiter: Enter pair delimiter

Fig. 05: Example Pipeline JSON.

To import the example Pipeline directly, copy and save the JSON below, then follow these instructions:

```
Zscaler Example Pipeline
{
  “id”: “zscaler”,
  “conf”: {
    “output”: “default”,
    “groups”: {},
    “asyncFuncTimeout”: 1000,
    “functions”: [
      {
        “id”: “code”,
        “filter”: “true”,
        “disabled”: false,
        “conf”: {
          “maxNumOfIterations”: 5000,
          “code”: “let temp = {};\n\n// Substr drops the timestamp from _raw, otherwise the
split does not work correctly\n__e[‘_raw’].substr(20).split(‘\\t’).forEach((element) => {\n
// Split K=V on the first equal sign\n    let eq = element.indexOf(‘=’)\n    let name = element.
substr(0, eq);\n    let value = element.substr(eq + 1);\n\n    // if value is none or N/A,
drop the field\n    value !== ‘None’ && value !== ‘NA’ ? temp[name] = value : false;\n    //
otherwise use this line below\n    // temp[name] = value;\n})\n\n__e[‘_raw’] = temp;”
        }
      },
      {
        “id”: “eval”,
        “filter”: “true”,
        “disabled”: false,
        “conf”: {
          “add”: [
            {
              “name”: “_raw.hostname”,
              “value”: “_raw.url.startsWith(_raw.hostname) ? undefined : _raw.hostname”
            },
            {
              “name”: “_raw.reason”,
              “value”: “_raw.reason === _raw.action ? undefined : _raw.reason”
            },
            {
              “name”: “_raw.bwthrottle”,
              “value”: “_raw.bwthrottle === ‘NO’ ? undefined : _raw.bwthrottle”
            }
          ]
        }
      },
      {
        “id”: “serialize”,
        “filter”: “true”,
        “disabled”: false,
        “conf”: {
          “type”: “kvp”,
          “fields”: [
            “!vendor”,
            “!product”,
            “!useragent”,
            “!location”,
            “!responsesize”,
            “!requestsize”,
            “!event_id”,
            “!*transtime”,
            “!transactionsize”,
            “*”
          ],
          “dstField”: “_raw”,
          “cleanFields”: false,
          “srcField”: “_raw”
        }
      }
    ]
  }
}
```

ABOUT CRIBL

Cribl, the Data Engine for IT and Security, empowers organizations to transform their data strategy. Customers use Cribl’s vendor-agnostic solutions to analyze, collect, process, and route all IT and security data from any source or in any destination, delivering the choice, control, and flexibility required to adapt to their ever-changing needs. Cribl’s product suite, which is used by Fortune 1000 companies globally, is purpose-built for IT and Security, including Cribl Stream, the industry’s leading observability pipeline, Cribl Edge, an intelligent vendor-neutral agent, Cribl Search, the industry’s first search-in-place solution, and Cribl Lake, a turnkey data lake. Founded in 2018, Cribl is a remote-first workforce with an office in San Francisco, CA.

Learn more: www.cribl.io | Try now: Cribl sandboxes | Join us: Slack community | Follow us: LinkedIn and Twitter

DGDE-0001-EN-1-0424