corelight

OPEN NDR

NETWORK DETECTION & RESPONSE

ZEEK® LOGS

conn.log | IP, TCP, UDP, ICMP connection details

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp of first packet
uid	string	Unique identifier of connection
id	record conn_id	Connection's 4-tuple of endpoints
> id.orig_h	addr	IP address of system initiating connection
> id.orig_p	port	Port from which the connection is initiated
> id.resp_h	addr	IP address of system responding to connection request
> id.resp_p	port	Port on which connection response is sent
proto	enum	Transport layer protocol of connection
service	string	Application protocol ID sent over connection
duration	interval	How long connection lasted
orig_bytes	count	Number of payload bytes originator sent
resp_bytes	count	Number of payload bytes responder sent
conn_state	string	Connection state (see **conn.log > conn_state**)
local_orig	bool	Value=T if connection originated locally
local_resp	bool	Value=T if connection responded locally
missed_bytes	count	Number of bytes missed (packet loss)
history	string	Connection state history (see **conn.log > history**)
orig_pkts	count	Number of packets originator sent
orig_ip_bytes	count	Number of originator IP bytes (via IP total_length header field)
resp_pkts	count	Number of packets responder sent
resp_ip_bytes	count	Number of responder IP bytes (via IP total_length header field)
tunnel_parents	table	If tunneled, connection UID value of encapsulating parent(s)
orig_l2_addr	string	Link-layer address of originator
resp_l2_addr	string	Link-layer address of responder
vlan	int	Outer VLAN for connection
inner_vlan	int	Inner VLAN for connection

conn_state

A summarized state for each connection

S0	Connection attempt seen, no reply
S1	Connection established, not terminated (0 byte counts)
SF	Normal establish & termination (>0 byte counts)
REJ	Connection attempt rejected
S2	Established, Orig attempts close, no reply from Resp
S3	Established, Resp attempts close, no reply from Orig
RSTO	Established, Orig aborted (RST)
RSTR	Established, Resp aborted (RST)
RSTOS0	Orig sent SYN then RST; no Resp SYN-ACK
RSTRH	Resp sent SYN-ACK then RST; no Orig SYN
SH	Orig sent SYN then FIN; no Resp SYN-ACK ("half-open")
SHR	Resp sent SYN-ACK then FIN; no Orig SYN
OTH	No SYN, not closed. Midstream traffic. Partial connection.

history

Orig UPPERCASE, Resp lowercase

S	A **S**YN without the ACK bit set
H	A SYN-ACK ("**h**andshake")
A	A pure **A**CK
D	Packet with payload ("**d**ata")
F	Packet with **F**IN bit set
R	Packet with **R**ST bit set
C	Packet with a bad **c**hecksum
I	**I**nconsistent packets (e.g., SYN & RST)
G	Content **G**ap
Q	Multi-flag packet (SYN & FIN or SYN + RST)
T	Re**t**ransmitted packet
W	Packet with zero **w**indow advertisement
^	Flipped connection

dhcp.log | DHCP lease activity

FIELD	TYPE	DESCRIPTION
ts	time	Earliest time DHCP message observed
uids	table	Unique identifiers of DHCP connections
client_addr	addr	IP address of client
server_addr	addr	IP address of server handing out lease
client_port	port	Client port at time of server handing out IP
server_port	port	Server port at time of server handing out IP
mac	string	Client's hardware address
host_name	string	Name given by client in Hostname option 12
client_fqdn	string	FQDN given by client in Client FQDN option 81
domain	string	Domain given by server in option 15
requested_addr	addr	IP address requested by client
assigned_addr	addr	IP address assigned by server
lease_time	interval	IP address lease interval
client_message	string	Message with DHCP_DECLINE so client can tell server why address was rejected
server_message	string	Message with DHCP_NAK to let client know why request was rejected
msg_types	vector	DHCP message types seen by transaction
duration	interval	Duration of DHCP session
client_chaddr	string	Hardware address reported by the client
msg_orig	vector	Address originated from msg_types field
client_software	string	Software reported by client in vendor_class
server_software	string	Software reported by server in vendor_class
circuit_id	string	DHCP relay agents that terminate circuits
agent_remote_id	string	Globally unique ID added by relay agents to identify remote host end of circuit
subscriber_id	string	Value independent of physical network connection that provides customer DHCP configuration regardless of physical location

dns.log | DNS query/response details

FIELD	TYPE	DESCRIPTION
ts	time	Earliest timestamp of DNS protocol message
uid & id		Underlying connection info > **See conn.log**
proto	enum	Transport layer protocol of connection
trans_id	count	16-bit identifier assigned by program that generated DNS query
rtt	interval	Round trip time for query and response
query	string	Domain name subject of DNS query
qclass	count	QCLASS value specifying query class
qclass_name	string	Descriptive name query class
qtype	count	QTYPE value specifying query type
qtype_name	string	Descriptive name for query type
rcode	count	Response code value in DNS response
rcode_name	string	Descriptive name of response code value
AA	bool	Authoritative Answer bit: responding name server is authority for domain name
TC	bool	Truncation bit: message was truncated
RD	bool	Recursion Desired bit: client wants recursive service for query
RA	bool	Recursion Available bit: name server supports recursive queries
Z	count	Reserved field, usually zero in queries and responses
answers	vector	Set of resource descriptions in query answer
TTLs	vector	Caching intervals of RRs in answers field
rejected	bool	DNS query was rejected by server
auth	table	Authoritative responses for query
addl	table	Additional responses for query

dpd.log | Dynamic protocol detection failures

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when protocol analysis failed
uid & id		Underlying connection info > **See conn.log**
proto	enum	Transport protocol for violation
analyzer	string	Analyzer that generated violation
failure_reason	string	Textual reason for analysis failure
packet_segment	string	Payload chunk that most likely resulted in protocol violation

files.log | File analysis results

FIELD	TYPE	DESCRIPTION
ts	time	Time when file first seen
fuid	string	Identifier associated with single file
uid & id		Underlying connection info > **See conn.log**
source	string	Identification of file data source
depth	count	Value to represent depth of file in relation to source
analyzers	table	Set of analysis types done during file analysis
mime_type	string	Mime type, as determined by Zeek's signatures
filename	string	Filename, if available from file source
duration	interval	Duration file was analyzed for
local_orig	bool	Indicates if data originated from local network
is_orig	bool	If file sent by connection originator or responder
seen_bytes	count	Number of bytes provided to file analysis engine
total_bytes	count	Total number of bytes that should comprise full file
missing_bytes	count	Number of bytes in file stream missed
overflow_bytes	count	Number of bytes in file stream not delivered to stream file analyzers
timedout	bool	If file analysis timed out at least once
parent_fuid	string	Container file ID was extracted from
md5	string	MD5 digest of file contents
sha1	string	SHA1 digest of file contents
sha256	string	SHA256 digest of file contents
extracted	string	Local filename of extracted file
extracted_cutoff	bool	Set to true if file being extracted was cut off so whole file was not logged
extracted_size	count	Number of bytes extracted to disk
entropy	double	Information density of file contents

ftp.log | FTP request/reply details

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when command sent
uid & id		Underlying connection info > **See conn.log**
user	string	Username for current FTP session
password	string	Password for current FTP session
command	string	Command given by client
arg	string	Argument for command, if given
mime_type	string	Sniffed mime type of file
file_size	count	Size of file
reply_code	count	Reply code from server in response to command
reply_msg	string	Reply message from server in response to command
data_channel	record FTP:: Expected Data Channel	Expected FTP data channel
fuid	string	File unique ID

http.log | HTTP request/reply details

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when request happened
uid & id		Underlying connection info > **See conn.log**
trans_depth	count	Pipelined depth into connection
method	string	Verb used in HTTP request (GET, POST, etc.)
host	string	Value of HOST header
uri	string	URI used in request
referrer	string	Value of referer header
version	string	Value of version portion of request
user_agent	string	Value of User-Agent header from client
origin	string	Value of Origin header from client
request_body_len	count	Uncompressed data size from client
response_body_len	count	Uncompressed data size from server
status_code	count	Status code returned by server
status_msg	string	Status message returned by server
info_code	count	Last seen 1xx info reply code from server
info_msg	string	Last seen 1xx info reply message from server
tags	table	Indicators of various attributes discovered
username	string	Username if basic-auth performed for request
password	string	Password if basic-auth performed for request
proxied	table	All headers indicative of proxied request
orig_fuids	vector	Ordered vector of file unique IDs
orig_filenames	vector	Ordered vector of filenames from client
orig_mime_types	vector	Ordered vector of mime types
resp_fuids	vector	Ordered vector of file unique IDs
resp_filenames	vector	Ordered vector of filenames from server
resp_mime_types	vector	Ordered vector of mime types
client_header_names	vector	Vector of HTTP header names sent by client
server_header_names	vector	Vector of HTTP header names sent by server
cookie_vars	vector	Variable names extracted from all cookies
uri_vars	vector	Variable names from URI

irc.log | IRC communication details

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when command seen
uid & id		Underlying connection info > **See conn.log**
nick	string	Nickname given for connection
user	string	Username given for connection
command	string	Command given by client
value	string	Value for command given by client
addl	string	Any additional data for command
dcc_file_name	string	DCC filename requested
dcc_file_size	count	DCC transfer size as indicated by sender
dcc_mime_type	string	Sniffed mime type of file
fuid	string	File unique ID

kerberos.log | Kerberos authentication

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
uid & id		Underlying connection info > **See conn.log**
request_type	string	Authentication Service (AS) or Ticket Granting Service (TGS)
client	string	Client
service	string	Service
success	bool	Request result
error_msg	string	Error message
from	time	Ticket valid from
till	time	Ticket valid until
cipher	string	Ticket encryption type
forwardable	bool	Forwardable ticket requested
renewable	bool	Renewable ticket requested
client_cert_subject	string	Subject of client certificate, if any
client_cert_fuid	string	File unique ID of client cert, if any
server_cert_subject	string	Subject of server certificate, if any
server_cert_fuid	string	File unique ID of server cert, if any
auth_ticket	string	Ticket hash authorizing request/transaction
new_ticket	string	Ticket hash returned by KDC

mysql.log | MySQL

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
uid & id		Underlying connection info > **See conn.log**
cmd	string	Command that was issued
arg	string	Argument issued to command
success	bool	Server replied command succeeded
rows	count	Number of affected rows, if any
response	string	Server message, if any

pe.log | Portable executable

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
id	string	File id of this portable executable file
machine	string	Target machine file was compiled for
compile_ts	time	Time file was created
os	string	Required operating system
subsystem	string	Subsystem required to run this file
is_exe	bool	Is file an executable, or just an object file?
is_64bit	bool	Is file a 64-bit executable?
uses_aslr	bool	Does file support Address Space Layout Randomization?
uses_dep	bool	Does file support Data Execution Prevention?
uses_code_integrity	bool	Does file enforce code integrity checks?
uses_seh	bool	Does file use structured exception handing?
has_import_table	bool	Does file have import table?
has_export_table	bool	Does file have export table?
has_cert_table	bool	Does file have attribute certificate table?
has_debug_data	bool	Does file have debug table?
section_names	vector of string	Names of sections, in order

radius.log | RADIUS authentication attempts

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
uid & id		Underlying connection info > **See conn.log**
username	string	Username, if present
mac	string	MAC address, if present
framed_addr	addr	Address given to network access server, if present
tunnel_client	string	Address (IPv4, IPv6, or FQDN) of initiator end of tunnel, if present
connect_info	string	Connect info, if present
reply_msg	string	Reply message from server challenge
result	string	Successful or failed authentication
ttl	interval	Duration between first request and either Access-Accept message or an error

sip.log | SIP analysis

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when request happened
uid & id		Underlying connection info > **See conn.log**
trans_depth	count	Pipelined depth into request/response transaction
method	string	Verb used in SIP request (INVITE, etc)
uri	string	URI used in request
date	string	Contents of Date: header from client
request_from	string	Contents of request From: header[1]
request_to	string	Contents of To: header
response_from	string	Contents of response From: header[1]
response_to	string	Contents of response To: header
reply_to	string	Contents of Reply-To: header
call_id	string	Contents of Call-ID: header from client
seq	string	Contents of CSeq: header from client
subject	string	Contents of Subject: header from client
request_path	vector	Client message transmission path, extracted from headers
response_path	vector	Server message transmission path, extracted from headers
user_agent	string	Contents of User-Agent: header from client
status_code	count	Status code returned by server
status_msg	string	Status message returned by server
warning	string	Contents of Warning: header
request_body_len	count	Contents of Content-Length: header from client
response_body_len	count	Contents of Content-Length: header from server
content_type	string	Contents of Content-Type: header from server

[1] The tag= value usually appended to the sender is stripped off and not logged.

smtp.log | SMTP transactions

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when message was first seen
uid & id		Underlying connection info > **See conn.log**
trans_depth	count	Transaction depth if there are multiple msgs
helo	string	Contents of Helo header
mailfrom	string	Email addresses found in From header
rcptto	table	Email addresses found in Rcpt header
date	string	Contents of Date header
from	string	Contents of From header
to	table	Contents of To header
cc	table	Contents of CC header
reply_to	string	Contents of ReplyTo header
msg_id	string	Contents of MsgID header
in_reply_to	string	Contents of In-Reply-To header
subject	string	Contents of Subject header
x_originating_ip	addr	Contents of X-Originating-IP header
first_received	string	Contents of first Received header
second_received	string	Contents of second Received header
last_reply	string	Last message server sent to client
path	vector	Message transmission path, from headers
user_agent	string	Value of User-Agent header from client
tls	bool	Indicates connection switched to using TLS
fuids	vector	File unique IDs attached to message
is_webmail	bool	If message sent via webmail

snmp.log | SNMP messages

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp of first packet of SNMP session
uid & id		Underlying connection info > **See conn.log**
duration	interval	Amount of time between first packet belonging to SNMP session and latest seen
version	string	Version of SNMP being used
community	string	Community string of first SNMP packet associated with session
get_requests	count	Number of variable bindings in GetRequest/GetNextRequest PDUs seen for session
get_bulk_requests	count	Number of variable bindings in GetBulkRequest PDUs seen for session
get_responses	count	Number of variable bindings in Get-Response/Response PDUs seen for session
set_requests	count	Number of variable bindings in SetRequest PDUs seen for session
display_string	string	System description of SNMP responder endpoint
up_since	time	Time at which SNMP responder endpoint claims it's been up since

socks.log | SOCKS proxy requests

FIELD	TYPE	DESCRIPTION
ts	time	Time when proxy connection detected
uid & id		Underlying connection info > **See conn.log**
version	count	Protocol version of SOCKS
user	string	Username used to request a login to proxy
password	string	Password used to request a login to proxy
status	string	Server status for attempt at using proxy
request	record SOCKS:: Address	Client requested SOCKS address
request_p	port	Client requested port
bound	record SOCKS:: Address	Server bound address
bound_p	port	Server bound port

software.log | Software observed on network

FIELD	TYPE	DESCRIPTION
ts	time	Time at which software was detected
host	addr	IP address detected running the software
host_p	port	Port on which software is running
software_type	enum	Type of software detected (e.g., HTTP::SERVER)
name	string	Name of software (e.g., Apache)
version	record Software:: Version	Software version
unparsed_version	string	Full, unparsed version string found
url	string	Root URL where software was discovered

ssh.log | SSH handshakes

FIELD	TYPE	DESCRIPTION
ts	time	Time when SSH connection began
uid & id		Underlying connection info > **See conn.log**
version	count	SSH major version (1 or 2)
auth_success	bool	Authentication result (T=success, F=failure, unset=unknown)
auth_attempts	count	Number of authentication attempts observed
direction	enum	Direction of connection
client	string	Client's version string
server	string	Server's version string
cipher_alg	string	Encryption algorithm in use
mac_alg	string	Signing (MAC) algorithm in use
compression_alg	string	Compression algorithm in use
kex_alg	string	Key exchange algorithm in use
host_key_alg	string	Server host key's algorithm
host_key	string	Server's key fingerprint
remote_location	record geo_ location	Add geographic data related to remote host of connection

ssl.log | SSL handshakes

FIELD	TYPE	DESCRIPTION
ts	time	Time when SSL connection first detected
uid & id		Underlying connection info > **See conn.log**
version	string	SSL/TLS version server chose
cipher	string	SSL/TLS cipher suite server chose
curve	string	Elliptic curve server chose when using ECDH/ECDHE
server_name	string	Value of Server Name Indicator SSL/TLS extension
resumed	bool	Flag that indicates session was resumed
last_alert	string	Last alert seen during connection
next_protocol	string	Next protocol server chose using application layer next protocol extension, if present
established	bool	Flags if SSL session successfully established
ssl_history	string	SSL history showing which types of packets were received in which order. Client-side letters are capitalized, server-side lowercase.
cert_chain_fps	vector	All fingerprints for the certificates offered by the server
client_cert_chain_fps	vector	All fingerprints for the certificates offered by the client
subject	string	Subject of X.509 cert offered by server
issuer	string	Subject of signer of X.509 server cert
client_subject	string	Subject of X.509 cert offered by client
client_issuer	string	Subject of signer of client cert
sni_matches_cert	bool	Set to true if the hostname sent in the SNI matches the certificate, false if it does not. Unset if the client did not send an SNI.
request_client_certificate_authorities	vector	List of client certificate CAs accepted by the server
server_version	count	Numeric version of the server in the server hello
client_version	count	Numeric version of the client in the client hello
client_ciphers	vector	Ciphers that were offered by the client for the connection
ssl_client_exts	vector	SSL client extensions
ssl_server_exts	vector	SSL server extensions
ticket_lifetime_hint	count	Suggested ticket lifetime sent in the session ticket handshake by the server
dh_param_size	count	The diffie helman parameter size, when using DH
point_formats	vector	Supported elliptic curve point formats
client_curves	vector	The curves supported by the client
orig_alpn	vector	Application layer protocol negotiation extension sent by the client
client_supported_versions	vector	TLS 1.3 supported versions
server_supported_version	count	TLS 1.3 supported version
psk_key_exchange_modes	vector	TLS 1.3 Pre-shared key exchange modes
client_key_share_groups	vector	Key share groups from client hello
server_key_share_group	count	Selected key share group from server hello
client_comp_methods	vector	Client supported compression methods
sigalgs	vector	Client supported signature algorithms
hashalgs	vector	Client supported hash algorithms
validation_status	string	Certificate validation result for this connection
ocsp_status	string	OCSP validation result for this connection
valid_ct_logs	count	Number of different logs for which valid SCTs encountered in connection
valid_ct_operators	count	Number of different log operators for which valid SCTs encountered in connection

ssl_history

^	direction flipped	U	certificate_status
H	hello_request	A	supplemental_data
C	client_hello	Z	unassigned_handshake_type
S	server_hello	I	change_cipher_spec
V	hello_verify_request	B	heartbeat
T	NewSessionTicket	D	application_data
X	certificate	E	end_of_early_data
K	server_key_exchange	O	encrypted_extensions
R	certificate_request	P	key_update
N	server_hello_done	M	message_hash
Y	certificate_verify	J	hello_retry_request
G	client_key_exchange	L	alert
F	finished	Q	unknown_content_type
W	certificate_url		

syslog.log | Syslog messages

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when syslog message was seen
uid & id		Underlying connection info > **See conn.log**
proto	enum	Protocol over which message was seen
facility	string	Syslog facility for message
severity	string	Syslog severity for message
message	string	Plain text message

tunnel.log | Details of encapsulating tunnels

FIELD	TYPE	DESCRIPTION
ts	time	Time at which tunnel activity occurred
uid & id		Underlying connection info > **See conn.log**
tunnel_type	enum	Tunnel type
action	enum	Type of activity that occurred

weird.log | Unexpected network/protocol activity

FIELD	TYPE	DESCRIPTION
ts	time	Time when weird occurred
uid & id		Underlying connection info > **See conn.log**
name	string	Name of weird that occurred
addl	string	Additional information accompanying weird, if any
notice	bool	If weird was turned into a notice
peer	string	Peer that originated weird
source	string	The source of the weird, often an analyzer name

x509.log | X.509 certificate info

FIELD	TYPE	DESCRIPTION
ts	time	Current timestamp
fingerprint	string	Fingerprint of the certificate
certificate	record X509:: Certificate	Basic information about certificate
san	record X509:: Subject Alternative Name	Subject alternative name extension of certificate
basic_constraints	record X509:: Basic Constraints	Basic constraints extension of certificate
host_cert	bool	Indicates if this certificate was a end-host certificate, or sent as part of a chain
client_cert	bool	Indicates if this certificate was sent from the client
cert	string	Base64 encoded X.509 certificate

FREE THREAT HUNTING GUIDE

Get Corelight's Threat Hunting Guide, based on the MITRE ATT&CK® Framework. Learn how to find dozens of adversary tactics and techniques using Corelight network evidence. Visit corelight.com or email info@corelight.com.

corelight

THREAT HUNTING GUIDE

How to threat hunt with Open NDR + MITRE ATT&CK®

Archive Collected Data
Automated Collection
Automated Exfiltration
BITS Jobs
Brute Force
Command Line Interface PowerShell
Commonly Used Ports/Non-Standard Ports
Data from Network Shared Drive
Data Transfer Size Limits
Drive-By Compromise
Encrypted Channel
External Remote Services
Fallback Channels, Multi-Stage Channels
Forced Authentication
Ingress Tool Transfer
Install Root Certificate
Network Sniffing
Network Service Scanning
Network Share Discovery
Non-Application Layer Protocol
Non-Standard Ports
Port Knocking
Proxy
Remote Desktop Protocol
Remote Services
Remote System Discovery
Server Software Component: Web Shell
Spearphishing Attachment
Spearphishing Link
Web Service
Windows Admin Shares

MICROSOFT LOGS

dce_rpc.log | Details on DCE/RPC messages

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
uid & id		Underlying connection info > **See conn.log**
rtt	interval	Round trip time from request to response
named_pipe	string	Remote pipe name
endpoint	string	Endpoint name looked up from uuid
operation	string	Operation seen in call

ntlm.log | NT LAN Manager (NTLM)

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
uid & id		Underlying connection info > **See conn.log**
username	string	Username given by client
hostname	string	Hostname given by client
domainname	string	Domainname given by client
server_nb_computer_name	string	NetBIOS name given by server in a CHALLENGE
server_dns_computer_name	string	DNS name given by server in a CHALLENGE
server_tree_name	string	Tree name given by server in a CHALLENGE
success	bool	Indicates whether or not authentication was successful

rdp.log | Remote Desktop Protocol (RDP)

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when event happened
uid & id		Underlying connection info > **See conn.log**
cookie	string	Cookie value used by client machine
result	string	Status result for connection
security_protocol	string	Security protocol chosen by server
client_channels	vector	Channels requested by the client
keyboard_layout	string	Keyboard layout (language) of client machine
client_build	string	RDP client version used by client machine
client_name	string	Name of client machine
client_dig_product_id	string	Product ID of client machine
desktop_width	count	Desktop width of client machine
desktop_height	count	Desktop height of client machine
requested_color_depth	string	Color depth requested by client in high_color_depth field
cert_type	string	If connection is encrypted with native RDP encryption, type of cert being used
cert_count	count	Number of certs seen
cert_permanent	bool	Indicates if provided certificate or certificate chain is permanent or temporary
encryption_level	string	Encryption level of connection
encryption_method	string	Encryption method of connection
ssl	bool	Flag connection if seen over SSL

smb_files.log | Details on SMB files

FIELD	TYPE	DESCRIPTION
ts	time	Time when file was first discovered
uid & id		Underlying connection info > **See conn.log**
fuid	string	Unique ID of file
action	enum	Action this log record represents
path	string	Path pulled from tree that file was transferred to or from
name	string	Filename if one was seen
size	count	Total size of file
prev_name	string	If rename action was seen, this will be file's previous name
times	record SMB:: MAC-Times	Last time file was modified

smb_mapping.log | SMB mappings

FIELD	TYPE	DESCRIPTION
ts	time	Time when tree was mapped
uid & id		Underlying connection info > **See conn.log**
path	string	Name of tree path
service	string	Type of resource of tree (disk share, printer share, named pipe, etc)
native_file_system	string	File system of tree
share_type	string	If this is SMB2, share type will be included

ALERT LOGS

intel.log | Intelligence data matches

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp when data discovered
uid & id		Underlying connection info > **See conn.log**
seen	record Intel::-Seen	Where data was seen
matched	set [enum]	Which indicator types matched
sources	set [string]	Sources which supplied data that resulted in match
fuid	string	If file was associated with this intelligence hit, this is uid for file
file_mime_type	string	Mime type if intelligence hit is related to file
file_desc	string	Files 'described' to give more context
cif	record Intel::CIF	CIF

notice.log | Interesting events and activity

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp for when notice occurred
uid & id		Underlying connection info > **See conn.log**
fuid	string	File unique ID if notice related to a file
file_mime_type	string	Mime type if notice related to a file
file_desc	string	Files 'described' to give more context
proto	enum	Transport protocol
note	enum	Notice::Type of notice
msg	string	Human readable message for notice
sub	string	Human readable sub-message
src	addr	Source address, if no conn_id
dst	addr	Destination address
p	port	Associated port, if no conn_id
n	count	Associated count or status code
peer_descr	string	Text description for peer that raised notice, including name, host address and port
actions	set[e-num]	Actions applied to this notice
email_dest	set	The email address(es) where to send this notice
suppress_for	interval	Field indicates length of time that unique notice should be suppressed
remote_location	record geo_loca-tion	If GeoIP support is built in, notices have geographic information attached to them
dropped	bool	Indicate if $src IP address was dropped and denied network access

SURICATA

Corelight's Suricata® and Zeek logs link alerts and evidence to accelerate incident response

suricata_corelight.log

FIELD	TYPE	DESCRIPTION
ts	time	Timestamp of the Suricata alert
uid & id		Underlying connection info > **See conn.log**
alert.category	string	Type of attack being detected
alert.metadata	vector	All metadata keywords from signature in "name:value" format. Conveys info such as modification time, deployment location, etc.
alert.rev	integer	Revision number of signature
alert.severity	count	Seriousness of attack, with 1 being most severe
alert.signature	string	Human-readable description of the attack type
alert.signature_id	count	Numeric signature identifier
community_id	string	The community ID generated by Suricata, if community ID is configured
flow_id	count	The Suricata-assigned flow ID in which the alert occurred
metadata	vector of strings	Application layer metadata, if any, associated with the alert (for example, flowbits)
pcap_cnt	count	The PCAP record count, present when the packet that generated the alert originated from a PCAP field
retries	count	The number of retries performed to write this log entry. Used in diagnostic sessions.
service	string	The application protocol
suri_id	string	The unique ID for the log record
tx_id	count	The Suricata-assigned transaction ID in which the alert occurred

CORELIGHT COLLECTIONS

Corelight delivers a comprehensive suite of network security analytics that help organizations identify more than 75 adversarial TTPs across the MITRE ATT&CK® spectrum. These detections reveal known and unknown threats via hundreds of unique insights and alerts using machine learning, behavioral analysis, and signature-based approaches. The following Corelight Collections focus on our behavioral and statistical analyses and are organized by focus areas:

Entity Collection

The Corelight Entity Collection gives security teams powerful identification capabilities around applications, devices, services, certs, hosts, and more to help them comprehensively understand and defend their environment.

PACKAGE	DESCRIPTION
Known Entities	Extract, aggregate, summarize and log individual network entities, including hosts, devices, names, users, and domains
Local Subnets	Identify local IPv4/v6 space subnets, both public and private
Application Identification	Identify over 150 applications, including BitTorrent, DropBox, Facebook, TeamViewer, WhatsApp, and many more

C2 Collection

Identify command and control activity with over 50 unique insights and detections.

PACKAGE	DESCRIPTION
HTTP C2	Detect known families of malware that conduct C2 communications over HTTP, such as Empire, Metasploit, and Cobalt Strike
DNS tunneling	Detect DNS tunneling behavior as well as the presence of specific tunneling tools such as Iodine
ICMP tunneling	Detect ICMP tunneling behavior as well as the presence of specific tunneling tools such as ICMP Shell
Domain generation algorithms (DGAs)	Detect C2 traffic based on DNS activity from malware using domain generation algorithms
Meterpreter	Detect C2 activity from Metasploit's Meterpreter shell across HTTP and generic TCP/UDP traffic

Encrypted Traffic Collection

Combining observable elements like timestamps and packet sizes with known behavior of protocols, our encrypted traffic analytics offer a practical approach to visibility that lets you see and act on what matters.

PACKAGE	DESCRIPTION
Cert Hygiene	Identify risk indicators in your TLS traffic, such as newly minted certificates, expiring certificates, and the use of weak encryption keys
Encrypted DNS Server Detection	Detect DNS-over-HTTPS traffic
Encryption Detection	Track and log information related to unknown or unusual encryption methods
RDP Inference	Capture information and inferences about encrypted and unencrypted RDP connections through client, authentication, and behavioral inferences
SSH Inference	Generate inferences about SSH connections, such as keystrokes, file transfers, or authentication attempts
SSH Stepping Stones	Detect a series of intermediary hosts connected via SSH
VPN Insights	Identify and log VPN traffic, including over 300 unique protocols, and providers

For more info on Corelight's analytics and detections, visit corelight.com/products/analytics.

COMMUNITY ID

When processing flow data from a variety of monitoring applications (such as Zeek and Suricata), it's often desirable to pivot quickly from one dataset to another. While the required flow tuple information is usually present in the datasets, the details of such "joins" can be tedious, particularly in corner cases. The "Community ID" spec for flow hashing standardizes the production of a string identifier representing a given network flow to reduce pivots to simple string comparisons. Learn more at github.com/corelight/community-id-spec.

DISRUPT ATTACKS WITH NETWORK EVIDENCE