



XKEYSCORE

Appids & Fingerprints

xkeyscore@nsa



Basic Syntax

- Syntax is similar to C:

```
function('name', level, <optional info> ) = 'search terms and patterns';
```

- Two main functions -- *appid* and *fingerprint*:

```
appid('chat/icq', 8.5, wireshark='icq', chatproc='ICQ') =  
    /^[^o]icq/c and $icq;
```

```
fingerprint('fingerprint/phone/nokia/generic') =  
    'user-agent: nokia' or  
    'profile: http://nds.nokia.com/uaprof/n';
```



Naming Conventions

Appids are named using a pseudo directory convention:

/application_type/sub_type/name



Levels

Levels are 1.0 – 9.9 with lower numbers meaning higher priority.

This allows multiple signatures to match a piece of traffic, but only the most specific appid will be applied. For example:

appid('chat', 9.9) = ...

appid('chat/yahoo', 9.8) = ...

appid('chat/yahoo/incoming', 9.7) = ...

If a session matches all three signatures, the appid will be 'chat/yahoo/incoming' since that has the best priority.



Application Type

Third parameter is the *application type*; if missing, we use the appid name up to the first slash as the type

```
appid('http/response', 9.2, 'web') = ...
```

```
appid('chat/yahoo/incoming', 9.1) = ...
```



Basic Search Patterns

XKEYSCORE supports Boolean operations and regular expressions

Raw text must be encapsulated between single quotes

- *'search term'*

Terms can be combined with Boolean logic

- *'search term' and 'another term' and not 'defeat term'*
- *'search term' or 'another term'*



Example

```
appid('voip/sip/IMS', 6.0, wireshark='sip') =  
('via: sip' or 'v: sip') and 'cseq:' and (  
  'p-access-network-info:' or  
  'p-called-party-id:' or  
  'p-charging-vector:' or  
  'p-charging-vector-addresses:' or  
  'p-media-authorization:' or  
  'security-verify:' or  
  'proxy-authorization:' and 'scscf' or  
  'path:' and 'pcscf' or  
  'path:' and 'scscf'  
);
```



Binary and Regex Patterns

Binary patterns can be represented by putting a `\x` in front of each value:

```
'\xff\xff\x00\x02'
```

Or use the hex function:

```
hex('ff:ff:00:02')
```

Use slashes to enclose regular expressions:

```
/[^a-zA-Z0-9]BTE/
```




Case sensitivity

Keywords and regular expressions are NOT case sensitive by default.

Append a 'c' to request case-sensitive evaluation:

```
'keyword'c  
/regex/c
```



Keyword Length

Keywords must be at least 3 characters or they will never hit. This minimum is increased to 4 at some sites for performance reasons.

Regular expressions must include a fixed "anchor" meeting the minimum keyword length.

Bad: `/[A-Z]{3}-[0-9]{3,5}/`

OK: `/ABC-[0-9]{3,5}/`



Appid vs Fingerprint

Each session gets *one* appid -- lowest level wins. It gets databased in the 'application' field.

All matching fingerprints are stored in the 'fingerprint' field. Level is ignored and can be omitted from fingerprint definitions.

Application Type*: 

Application Info*:

Application: 

AppID (+Fingerprints)* [\[fulltext\]](#):   [\[Field Builder\]](#)

Winning appid (red arrow pointing to Application field)

Winning appid + all fingerprints (blue arrow pointing to AppID field)



Example

```
appid('mail/yahoo', 9.0) = 'Host: mail.yahoo';  
appid('mail/yahoo/login', 8.0) = 'Host: mail.yahoo' and '/login';  
  
fingerprint('mail/arabic') = 'mail' and '/language[:=] ?ar/';  
fingerprint('mail/yahoo/ybm') = 'Host: mail.yahoo' and 'YBM='c';
```

```
GET /login.html HTTP/1.1  
Referer: http://us.f359.mail.yahoo.com/ym/ShowLetter  
Accept-Language: ar  
Accept-Encoding: gzip, deflate  
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)  
Host: mail.yahoo.com  
Connection: Keep-Alive  
Cookie: B=fn50ehd2612o2&b=3&s=rp; YBM=d=&v=1;
```

Application: mail/yahoo/login

Fingerprint: mail/yahoo/login mail/arabic mail/yahoo/ybm



Real world example

رسالة خاصة: تفصل ..

PM 08:04 ,2008-18-02

تاريخ التسجيل: Mar 2007
المشاركات: 1,199

tawab
مير متواجد حالياً tawab
نائب المشرف

افتراضي تفصل ..

Begin ASRAR EI Mojahedeen v2.0 Encrypted Message

r/RgTzT/ATRhN2E1Zjg1OWQyNWRjMmE2ZTdINzZmZDhIODUxZWZhMDQ1MjYwMjVIZGU0ZGYwMjdkMmJmNTA4ZDY2Yjk0MGU2NGNYjg5MzNjZTc5MThjY2Y1ZmY5MTgzZDIkYjhjMTExOGYzYjc1ZDdiMDAxNTQzZmVINDVIY2YyMGJjYjU2ODkyYjdmYjFjYjAzMWM5ZDQ2OWFIMzg4NThhM2I1Mjc5ODkzZGNhOGRmNWJmNjVIZjQ0MjMxNDM4MDIyO Tg1MmRjMGJINGNkYTNkYTQ4MzMxZjRiN2FiNjI3MjE1NGI3MTA3ZDQ4NWRmYzMyOTUzZjZIMjg3NjQ1OGQ4MTA3NTU2N2Zkn2ZjYzUzYzYyMjFjODAwN2VhM2U5MTZiNDY2MmM2ZTVlYjQ2YzI0OGQ2ODUxNWVhMjI2MwViNDAY0GI0MThkMTdhNTY1YzIxMDgyOGZIM2lwZWZjMDgwM2U4MzNINDg1ODUxZTc4ODc1MTY2M2I0NjU5ZjBhZjVhNjk0OTIhNGExOThmYWVI NmFIZjlyNmMwZDA3MDM0NjJkZDhhMml4ZmRhYjc3NmZINDFkODkyYjBhYjY3MDQ1OGVIMjdhYmUwZTlyNGlxYmQyZDIzZjliM2E5ZGQ5NmNhZDQxOTM4NTI0Mjc3MzBIOWEwZWE1Njk3YjgxY2VINTQ1OWULnoiVDULIjTEuDJqneOGMRHesi/BPTnZj02yqbmKbFkIPjwMhe7FUhFAOw74S+i+P0kOREo5XhdP+y9/

Gul3juYTvrlEOxGx2OsSfNS5kfRXXH1DaTnb7Oyufe9r6mMIQ6e6E0SRUIdu6YVupz0hhgd4Dof

SBbFR3OvgOS+pUxDYgmEOr/RA+fYi47tuHQMh+dynZqQspNdmRUmkjEpFqF03sPHS/10injQoe1Gsf8+xn52XE2q/WdnU+4XjWnl/isVNAjv2nsL+s2TG1HbgocmpQoxy0B0SXPcRv/+2JekV37k1XyONZk9YH+DV3aWYPXt+ym+wG0XNTqPHIU1JWAZql2NK/cSXt9DMtCtcb8czRj6G9IXvJ9Eny7t06xPd9BGio9M+3QuUkZHLEmJiAvgvB6R/X/3whBqk6zMHQLfo+VJcX9umW5mRtgCjzSPW6lzzFCGt/B4SK4PxT52ZC0B2kWD8VMYnffrlsTG4XUesgx47Nd5xML8w5pj/fZwKNK+EfkIP==Z1ow29A9N3uLIXBX62LhOyj/1iqfJ2FNR7AIONSEjwKoggVmkxDiuGaQi+TurpxBgat1g

End ASRAR EI Mojahedeen v2.0 Encrypted Message

Displaying 1 items
Hidden fields

لوحة التحكم

لإعدادات والخيارات

[تعديل التوقيع](#)

[البريد الإلكتروني / كلمة المرور](#)

[تعديل الملف الشخصي](#)

[تعديل الخيارات](#)

لرسائل الخاصة

[قائمة الرسائل](#)

[إرسال رسالة جديدة](#)

[تتبع الرسائل](#)

[تعديل المحادثات](#)

مواضيع مشتركة بها

[قائمة الاشتراكات](#)

[تعديل المحادثات](#)

لرقابة

[المواضيع المحذوفة](#)

[المشاركات المحذوفة](#)

[المواضيع المراقبة](#)

[المشاركات المراقبة](#)

لمنتجات

[التذكير بالحدث](#)

[قوائم لأصدقاء / النجاهل](#)

[الملفات المرفقة](#)



Distribution

- Appids and fingerprints are distributed across the XKEYSCORE network every hour
- Changes will take effect within 2 hours of check-in
- Current definitions are available on the website:

<http://xkeyscore.r1.r.nsa/documents/appid.html>



Intermediate Syntax



Append Option

You can append derived metadata fields onto the end of an appid:

```
appid('p2p/kazaa', 7.7, append='mime_type') =  
lpos('x-kazaa') and not $http;
```

This will result in an appid like 'p2p/kazaa/image/jpeg'.



Built in functions

ip(expr)	<p>Matches against an IP Address looks in to address and from address in the session headers</p> <ul style="list-style-type: none"> • ip('10.10.10.1');
toport(expr)	<p>Matches against the Destination/To port. Note this must be a numeric representation of a port.</p> <ul style="list-style-type: none"> • toport(1920);
fromport(expr)	<p>Matches against the Source/From port. Note this must be a numeric representation of a port.</p> <ul style="list-style-type: none"> • fromport(80);
port(expr)	<p>Matches against the either port. Note this must be a numeric representation of a port.</p> <ul style="list-style-type: none"> • port(6667);
next_protocol(expr)	<p>Matches against the integer version of the next protocol.</p> <ul style="list-style-type: none"> • next_protocol(250);
protocol ('text')	<p>Will only work for IP next protocol names as defined in the IANA next protocol numbers document</p> <ul style="list-style-type: none"> • protocol('tcp');



Built in functions

email_address(sel)	permutes just like strong_selector (just like DECODEORDAIN)
mac_address(addr)	Tasks a mac address
smac(addr)	
dmac(addr)	
ip(addr)	tasks this IP address (either to or from)
from_ip(addr)	tasks this IP address only when it is the originator
to_ip(addr)	tasks this IP address only when it is the destination



More built in functions

first(expr)	Matches against a pattern at the beginning of the session
lpos(expr)	Matches against a pattern at the beginning of each line (\n)
pos(expr)	expression occurs at offset X in the session <ul style="list-style-type: none"> • pos('Hello') == 5, • pos(/Good.*Grief/) <= 10
between(expr)	<ul style="list-style-type: none"> • between('Hello', 'World', 10, 100) <p>Separation between 'Hello' and 'World' is greater than or equal to 10 bytes and less than or equal to 100 bytes</p> <p>This is the same as using the following regular expression:</p> <ul style="list-style-type: none"> • /Hello.{10,100}World/
'term'c	Does a case sensitive match of the term
'term'u	Treats the term as UTF-16



Example

```
appid('voip/skinny(port2000)', 9.9, wireshark='skinny') =  
port(2000);
```

```
appid('voip/skinny/keep-alive', 3.0, wireshark='skinny') =  
toport(2000) and  
first('\x04\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00');
```

```
appid('voip/skinny/keep-alive-ack', 3.0, wireshark='skinny') =  
fromport(2000) and  
first('\x04\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00');
```



Example

```
appid('mail/smtp/to_server', 8.5, direction=$from_server,  
      wireshark='smtp') =  
      toport(25) and  
      ( first('helo') or  
        first('ehlo') or  
        first('data') or  
        (lpos('To: 'c) and lpos('From:'c)) or  
        lpos('QUIT'c) or  
        lpos('mail from:') or  
        lpos('rcpt to:') );
```



CHAINWORDS

You can assign a pattern to a variable (CHAINWORD) and reuse the variable in many patterns.

```
$sip = 'via: sip' and 'cseq:' and 'SIP/2'c;
```

Now we can use this variable in future definitions:

```
appid('voip/sip', 7.2 ) = $sip;
```

```
appid('voip/sip/invite', 6.9) = $sip and 'INVITE';
```



Predefined Chainwords

There are a number of chainwords predefined for convenience:

- \$tcp
- \$udp
- \$icmp
- \$sctp
- \$rpc
- \$arp
- \$ssl
- \$http_cmd
- \$http
- \$http_get
- \$http_put
- \$http_post
- \$http_delete
- \$http_trace
- \$http_head
- \$http_options
- \$http_partial
- \$vbulletin
- \$mime_type
- \$user_agent



Example

```
$icq = 'ICQ'c and $http and not (port(80) or $html_body or  
$http_cmd);
```

```
appid('chat/icq', 8.5, wireshark='icq', chatproc='ICQ') =  
/[^o]icq/c and $icq;
```

```
appid('chat/icq', 9.0, wireshark='icq', chatproc='ICQ') =  
first('icq') and not port(25);
```




Context sensitivity

Expressions are evaluated only with a certain context instead of across the session as a whole.

`html_title('Yahoo! Mail' or 'Yahoo! Address Book')`

... only hits if those keywords are seen within the title of a web page

`http_host('maps.google.com')`

... only hits within the "Host:" HTTP header



Context sensitivity

Why use context-sensitive scanning?

- More intuitive - you can say what you mean
- More accurate - if 'maps.google.com' is mentioned in a blog post, you don't want to try processing it as a Google Maps session
- Better performance for XKEYSCORE



Context sensitivity

Sample contexts:

html_title	filename
url	file_ext
http_host	doc_title
http_referer	doc_subject
http_cookie	doc_author
http_server	doc_org
user_agent	doc_hash
web_search	doc_body
to_cc	email_body
from_cc	chat_body



Example

```
appid('finance/currency_conversion/generic', 8.0) =  
  html_title('currency' and ('exchange' or 'conver')) or  
  http_server('currency' or 'x-rates.com');
```

```
appid('finance/currency_conversion/xe', 8.0) =  
  http_host('xe.com') or  
  html_title(/^XE -/c or 'XE.com'c);
```



Appid utility

appid options:

- help this help message
- list-all list all the application/fingerprint names and levels
- list-appids list all the application names (no fingerprints)
- list-fingerprints list all the application names (no appids)
- list-types list all the application types
- list-levels list all the application levels
- unit-test perform unit tests with data in the heirachy 'datadir', with files matching 'filespec'
- quiet don't print any load messages
- appid_fname arg location of appid.cfg
- input-file arg input file to test
- datadir arg The test data directory. Defaults to \$(XSCORE_TEST_DATA_DIR)/appids
- filespec arg (=.*\u124) A regular expression to match against files to check
- noexit arg (=0) do not stop on the first error



Appid Validation

appid sample.u124

Loading appids

- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/appid_definitions.cfg
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/anonymizer.appid
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/bulletin_board.appid
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/tao_vpn.appid
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/tdmoip.appid
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/terminal.appid
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/voip.appid
- >Loading : /home/oper/xkeyscore/config/dictionaries/appid/appid_definitions.cfg

Finished loading appids

Filename: sample.u124

Appid: encryption/https

Total Size: 19.36Kbits

Total Time: 0.01secs

Rate: 1.936Mbits/s

Overall performance:

Total Time: 0.01secs

Total Bits: 0.01936Mbits

Overall Rate: 1.936Mbits/s



Advanced Syntax



Code-based appids

Keywords and regular expressions don't work for everything

- Looking down columns in packet data
- Checksums
- Decoding (urlencoding, base64, gzip, etc.)



Code-based appids

Basic idea:

1. Preliminary "trigger" using standard keywords and regular expressions
2. Secondary test using a snippet of C++ code



Code-based appids

Example -- verifying a length field:

```
appid('netmanagement/ospf', 2, wireshark='ospf') =  
  protocol('ospf')  
: c++ {{  
  
  if (size() < 4)  
    return false;  
  
  const uint8_t *data = begin();  
  return (data[3]==size());  
}};
```



Code-based appids

Example -- packet data:

```
fingerprint('vpn/xxp_example') =  
  'Next Protocol 250'  
  : c++ [{  
    packet_t pkt;  
    int count = 0;  
    while ((pkt = get_packet()) && count < 20) {  
      ++count;  
      if (pkt.size < 16)  
        return false;  
  
      if (pkt.data[4] != 0xCC ||  
          pkt.data[5] != 0x45 ||  
          pkt.data[15] != 0x72)  
        return false;  
    }  
    return (count > 0);  
  }];
```



Context-sensitive code

Example -- code-based check on certain extracted files:

```
fingerprint('crazy/office') =  
  extracted_file('doc' or 'xls' or 'ppt' : c++ {{  
    return xks::filename().find("crazy") != std::string::npos;  
  }});
```



Extractors

Simplified regex-based metadata extraction

```
fingerprint('maps/google/example') =  
  http_host('.google.') and url('/maps?')  
: c++  
extractors : {{  
  ka = /Keep-Alive: (\d+)/;  
  accept[] = /Accept-([\^:]+): ([\w-]+)/;  
}}  
main : {{  
  if(ka && ka[0] == "300") {  
    for(size_t i = 0; i < accept.size(); ++i)  
      if(accept[i][0] == "Encoding" && accept[i][1] == "gzip")  
        return true;  
  }  
  return false;  
}};
```



Flex

Support for flex-based pattern matching

```
fingerprint('maps/google/firefox') =  
  http_host('.google.') and url('/maps?')  
: c++  
flex : {{  
  USER_AGENT_CHAR [^\n\r]  
  %%  
  "User-Agent: "{USER_AGENT_CHAR}+  {  
    std::string agent(ytext);  
    if(agent.find("Firefox") != std::string::npos)  
      return true;  
  }  
}};
```



Microplugins

The next step: giving code-based appids (limited) access to the XKS core

- Accessing top-level session metadata
- Throwing common events
- Contributing metadata for databasing

The goal: higher level of agility with lower learning curve



Microplugins

Example: accessing session metadata

```
fingerprint('maps/google/server1') =  
  http_host('.google.') and url('/maps?')  
: c++  
main : {{  
  return (SESSION["to_ip"] == "123.45.67.1");  
}};
```




Microplugins

Example: throwing a document_metadata event

```
fingerprint('maps/google/contrived') =  
  http_host('.google.') and url('/maps?')  
: c++  
main : {{  
  xks::doc_meta_t dm;  
  dm.filename = "google.txt";  
  dm.author = "Google, Inc.";  
  xks::document_metadata(dm);  
}};
```



Microplugins

Example: contributing metadata to HTTP Activity

```
fingerprint('maps/google/search') =  
  http_host('.google.') and url('/maps?')  
: c++  
extractors : {{  
  q = /[&?]q=([^&]+)/;  
}}  
main : {{  
  if(q) {  
    DB["http_parser"]["search_terms"] = xks::urldecode(q[0]);  
    DB.apply();  
    return true;  
  }  
}};
```