

**Installation documentation for Pepper Jack's  
OS-Type Detection (with p0f), Time-of-Day (TOD),  
Day-of-the-week (DOW), and BastardList plug-ins for Snort.**

----- Part One, Installation -----

Compilation and installation ( the easy way for most people ):

download the patched snort 2.6.1.3 source tree:

```
wget www.autoshun.org/downloads/snort-2.6.1.3-  
Patched_2_perfection.tgz
```

Rebuild snort from source.

```
tar -xzf snort-2.6.1.3-Patched_2_perfection.tgz  
cd snort-2.6.1.3-all-jp  
./configure --with-mysql && make
```

Refer to normal snort manuals for additional steps.

Compilation and installation ( the not so easy way, but necessary for some people ) :

Download the patchfile:

```
wget www.autoshun.org/downloads/JacksPatch-2.6
```

Patch your (pre existing) snort source tree:

```
cd /usr/src/ snort-2.6.1.3  
patch -p1 < JacksPatch-2.6
```

Rebuild snort from source.

```
make distclean [ error messages are OK, here ]  
./configure --with-mysql && make
```

----- Part Two, Writing better Snort rules -----

Using the TOD plug-in:

The **TOD** plugin makes All times are expressed in the Local Time of the snort sensor. It requires two Time-Of-Day values based on a 24 hour clock. It returns true if the current time is in between the two hour settings. Times are expressed as hours on a 24 hour clock. Times are inclusive. Hour 24 will be automatically corrected to the more technically correct "0".

Some Simple Examples:

```
time1 time2 current_time ResultMatch
```

6	18	12noon	true
6	18	18:59	true ( 0600 – 1859 )
6	4	4:00	false ( 0600 – 0459 = 23 hours )
6	4	12:30	true
6	18	20 (8pm)	false
6	4	2:20	true
6	18	2:20	false
4	4	4	true ( from 0400 – 0459 )

Negation is also supported by adding the "not" value. Some sample time of day elements:

```

tod: 4-6;
tod: 6-18;
tod: 4;
tod: !6-18;

```

multiple "tod;" entries can occur on a single rule but that will really slow things down. It is better to stack the time requirements into a single "tod:" element. Multiple time intervals in a single "tod:" element will be "OR"ed together. Multiple "tod:" elements will be "AND"ed together. Only one negation is allowed per "tod:" element and it negates the entire "tod:" element.

Some complex examples:

"hit only if the time is between 6am - 12noon or between 6pm - 12midnight" =>  
 tod: 6-11,18-24; ( 6:00am -> 11:59am or 6:00pm -> 11:59pm ).

"hit only if the time is between 8am - 5pm" => tod: 8-16; ( goes to 16:59)

"hit only if the time is not between 8am - 5pm" => tod: !8-16;

"hit only if the time is between 8pm - 5am" => tod: 20-4; ( goes to 04:59 )

"odd number hours" => tod: 1,3,5,7,9,11,13,15,17,19,21,23;

### Using the DOW plug-in:

The **DOW** plugin makes certain rules valid only on certain days of the week:

sunday=0, saturday=6. All times are expressed in the Local Time of the snort sensor.

Example: To express "this rule only hits on Wednesday" use the element: "dow: 3;" .  
Multiple days can be expressed like this: "dow: 0,6;" (Weekends) or "dow: 1,2,3,4,5;" (US Workdays)

### Using the "**Bastard List**" plug-in:

The **Bastard List** plug-in applies a "badness" score to an IP address. Using the Bastard List plugin requires a binary file from afferentsecurity.com. It must be installed on the sensor as "/etc/affblist.bin". The binary file includes a score for a given address. Every address has a score from 0 to 100. 100 means that the address in question is certainly an evil address. 0 means that the address is clean. It is a probability score based on some proprietary scoring technology from @fferent Security Labs (ASL). We use the scores with the permission of ASL and subject to the licence they have attached to the scoring technology and the binary file format.

Downloading the binary "bastard List" file is free but does require a registration with ASL. We have a short cut link to the @fferent Security Labs registration on the [www.autoshun.org](http://www.autoshun.org) web site.

The bastard list plugin uses a relational operator and a numerical value. The valid relational operators are:

- gt => greater than
- ge => greater than or equal to
- eq => equal to
- le => less than or equal to
- lt => less than

so a bastard element in a rule might look like this: `bastard: src,ge,60;` or `bastard: dst,le,50;`

You get the idea: simple stuff that may help turn a good snort rule into a great rule. For instance, here is a good rule that always catches the bad guys, but sometimes hits on our biz partners or 401k manager who mails to every employee in the entire company all at once.

```
bleeding-dos.rules: alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg: "BLEEDING-EDGE DOS Excessive SMTP MAIL-FROM DDoS"; flow: to_server, established; content:"MAIL FROM\:"; nocase; window: 0; id:0; threshold: type limit, track by_src, count 30, seconds 60; classtype: denial-of-service; sid: 2001795; rev:7; )
```

Here is the same rule modified to catch spambots by the dozens:

```
alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:
"BLEEDING-EDGE SpamBot sending 30 messages in 60 seconds"; flow: to_server,
established; content:"MAIL FROM:"; nocase; window: 0; id:0; threshold: type limit,
track by_src, count 30, seconds 60; classtype: denial-of-service; sid: 2001795; rev:8;
bastard: src,ge,70;)
```

-----

Downloading and Using the Autoshun plug-in is free but does require a registration with ASL. We have a short cut link to the @fferent Security Labs registration on the [www.autoshun.org](http://www.autoshun.org) web site. Using the Autoshun plug-in is quite a bit more complex. Full instructions are on the [www.autoshun.org](http://www.autoshun.org) site.