Denial of service

Suppose a python web server that implements (1) receives a file via HTTP, (2) process the file, and (3) sends back a reply via HTTP. In order to allow multiple concurrent users to use the server, the server use a dedicated thread for each HTTP request. Also, in order to reduce the size of data communicated via network, the server uses a last generation (I'm kidding) compression technology. A file consists of a list of substitution rules and a payload. Every line of the file that starts with `#`is considered a rule. For instance, the following file

```
# H = and

# AR = are

# brh = brotherhood

All human beings AR born free H equal in dignity H rights. They

AR endowed with reason H conscience H should act towards one

another in a spirit of brh.
```

can be decompressed to

```
All human beings are born free and equal in dignity and rights. They

are endowed with reason and conscience and should act towards one

another in a spirit of brotherhood.
```

File `printer.py` implements the function `decompress`.
This system has a vulnerability. An attacker can send a really small input and force the server to allocate a large amount of memory. Even worst, in some cases this amount of memory is unbounded. (Discuss with your colleagues additional problems related to the usage of one thread per request).

To fix this problem, you must complete the stub in `filter.py`, which returns `True` only for inputs that do not lead to unbound allocation of memory. Your filter will be used in the server to discard malicious requests.

To test your solution execute `./test.py` or `py.test test.py`.