Let's take this as our starting point:

"No matter how paranoid you are, what they're actually doing is worse than you can possibly imagine."

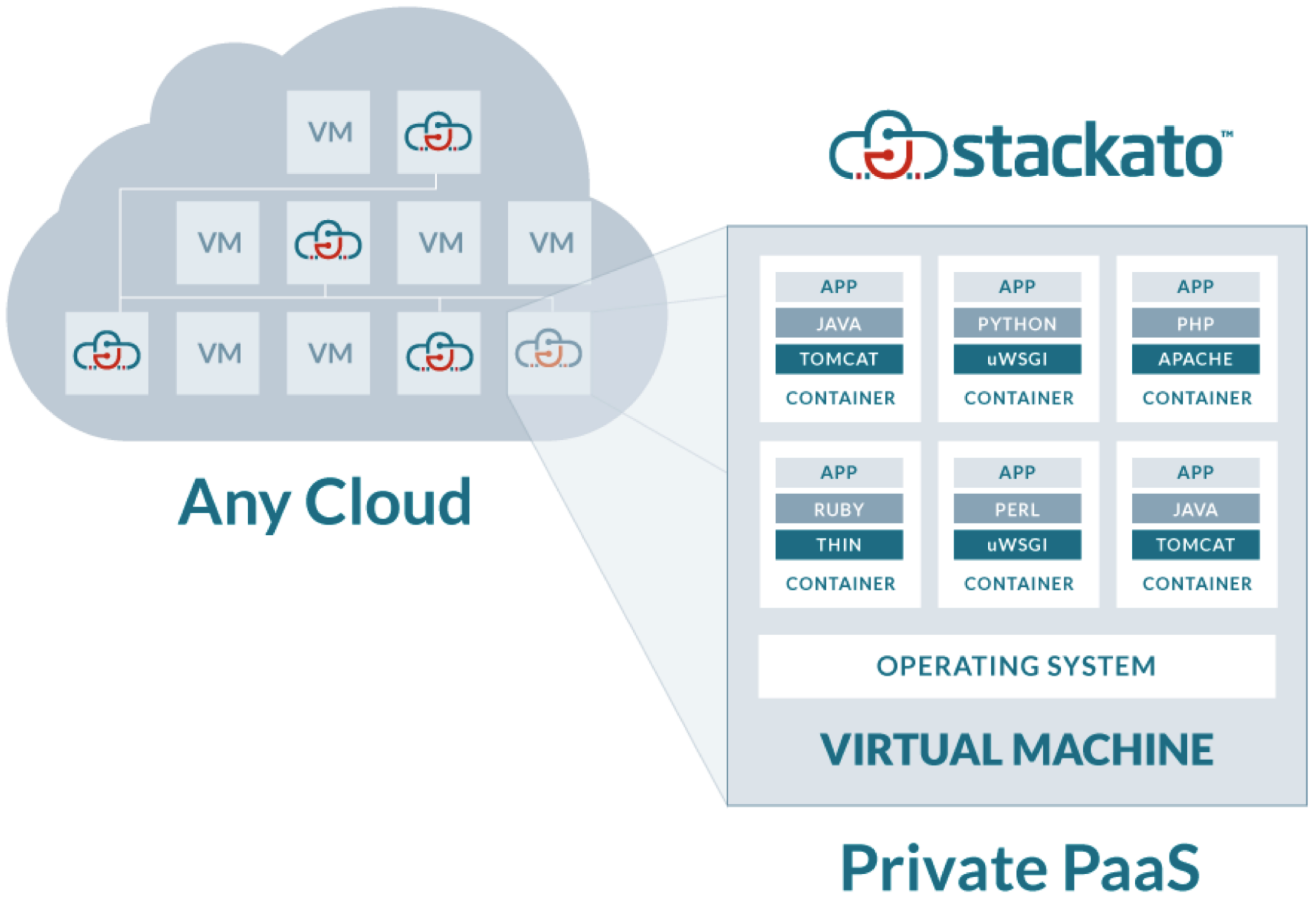
- Ralph J. Gleason (1917-75)

Or to be specific:

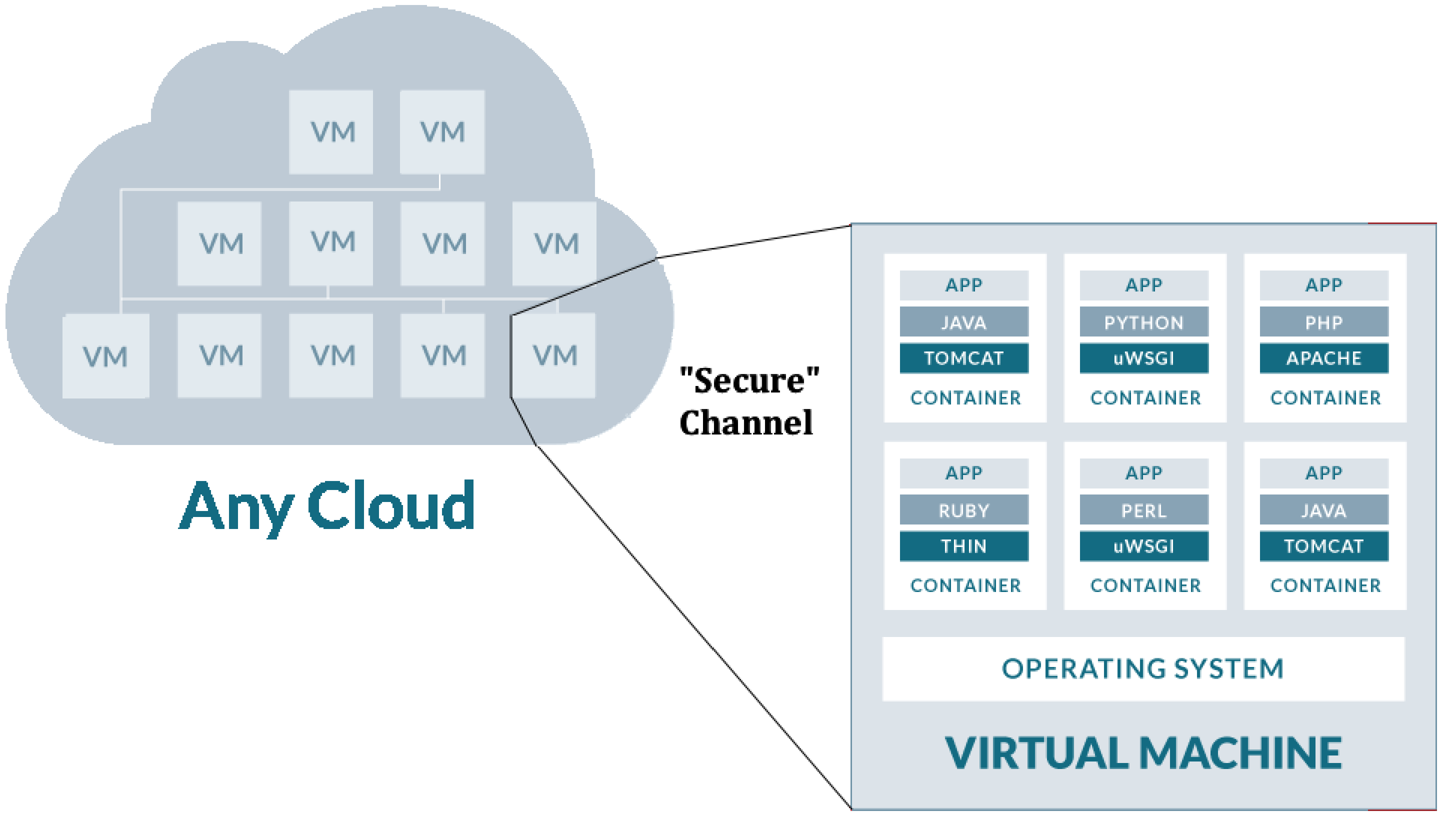
While misuse and abuse of the NSL power has been widely documented, the Obama administration [is seeking to allow] the FBI to demand even more records without court approval. [T]he administration proposed to expand the statute to allow the FBI to get American's internet activity records without court approval or even suspicion of wrongdoing.

http://www.aclu.org/national-security/doe-v-holder

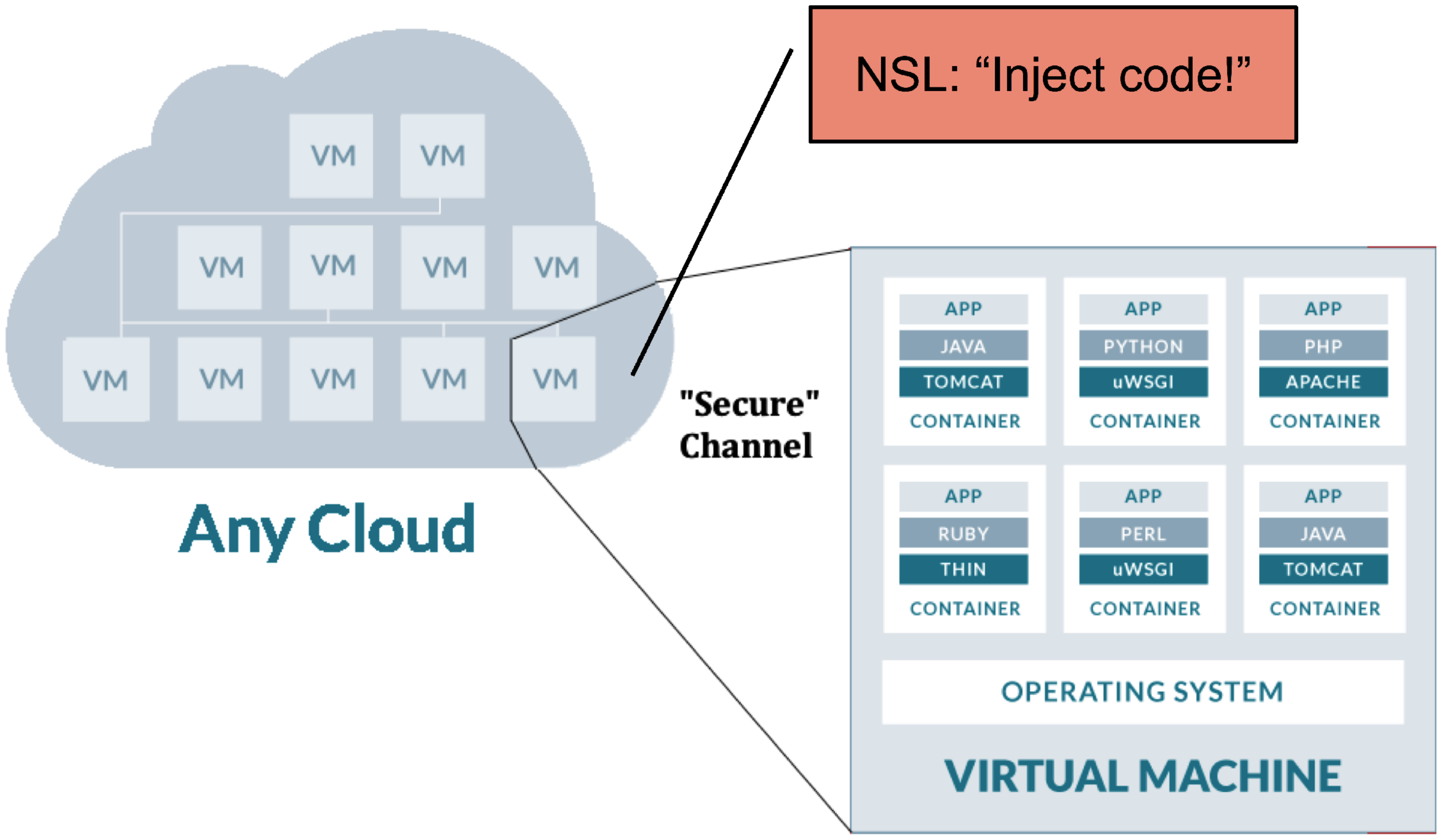
Let's take an illustration from ActiveState:



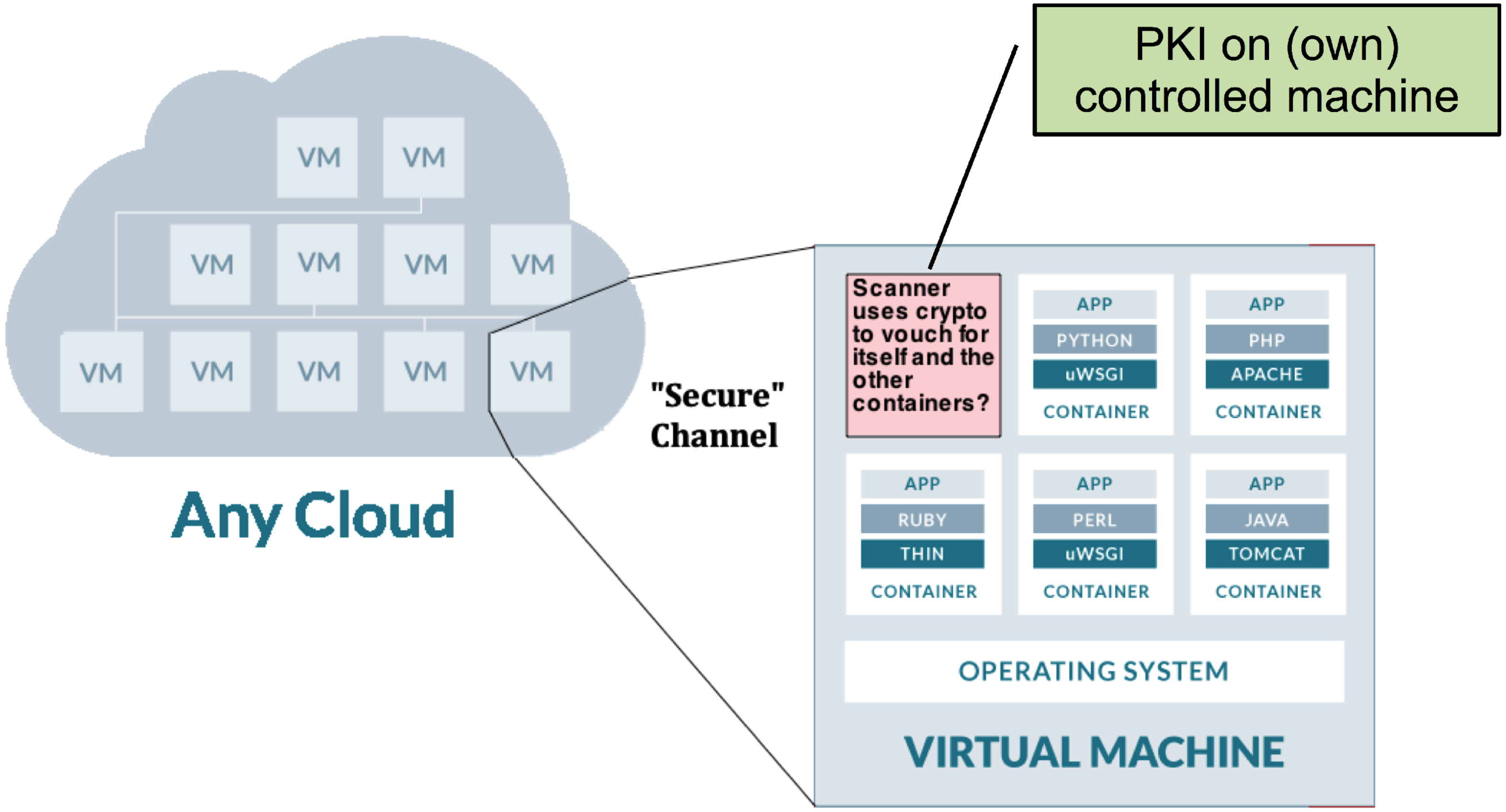
What security guarantee does this give us? It *does* verify that the bytes that make up the VM received by the Cloud Host are those you intended.



If Cloud Host receives a National Security Letter they might be compelled to inject code into your VM (and have a gag order against revealing they did so).

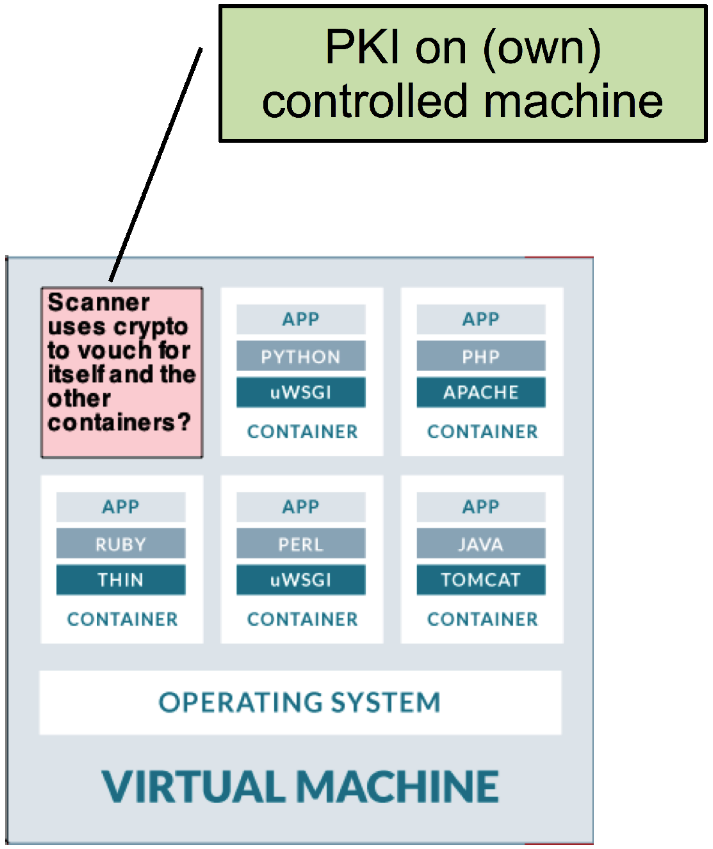


It would be *nice* if clever cryptography can let a process self-verify against code injection. But is it possible in the face of a bad actor or a hostile law?



If the "Scanner" can vouch for itself, and it can poke at the bytes inside other containers, this is sufficient to guarantee against injection attacks. How might it do this?

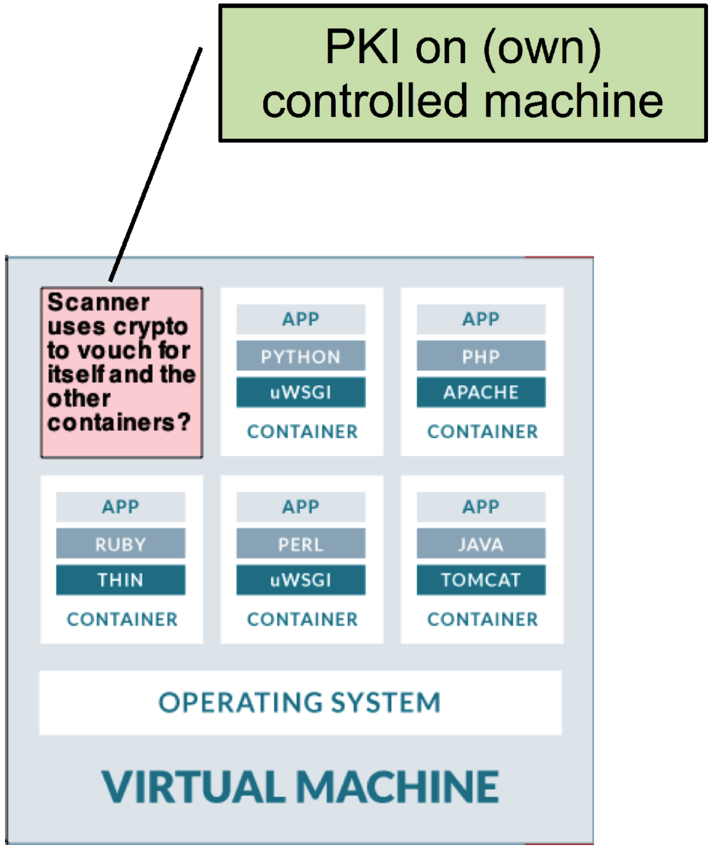
- Public key authentication against secured machine?
- Response to random queries of its own memory image?
- Response to random timing challenges to demonstrate known behavior?
- OS authentication of scanner? (but VM could inject into OS)



As can the audience, I can quickly poke holes in each of the methods in the last slide. On the other hand, I am not certain this quest is quixotic.

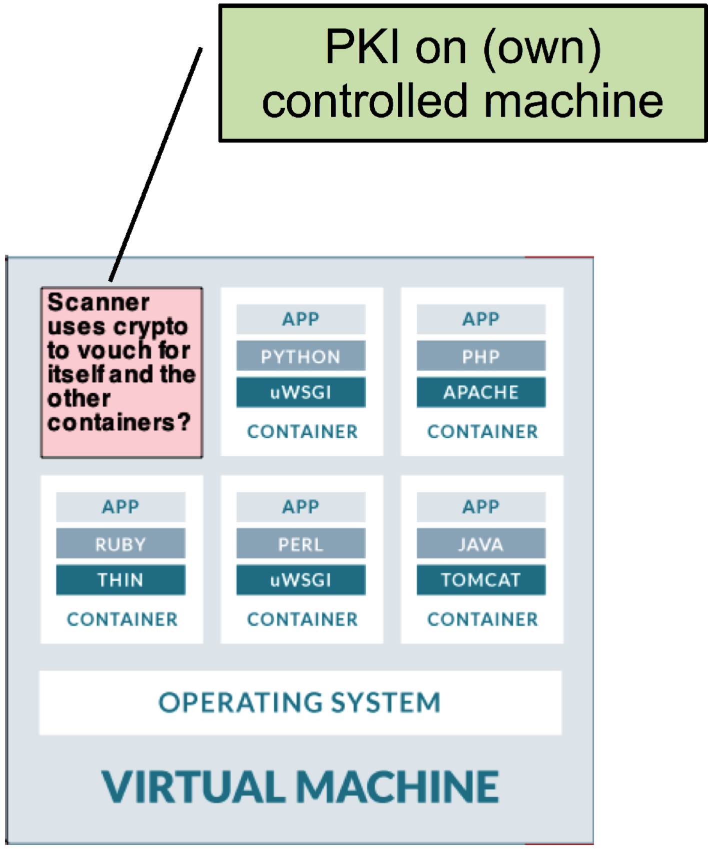
Inspirations:

- GPG/PGP: RSA lets me to send messages over insecure SMTP with assurance that only the intended recipient has access.
- Freenet: I can participate in a peer-to-peer data network without having even the *capability* of revealing or determining which content my node helps share.



Even if a "Scanner" is possible with the desired properties, it does nothing whatsoever to protect against attacks on applications within containers. App-level security is a distinct issue.

- If code running in a container is the binary intended (i.e. no injection), it may still needs to encrypt connections/stored data/etc. per app requirements.
- App-level software has known and unknown attacks. The best we are hoping for is "<u>no worse than</u>" hosting an application on privately controlled hardware.



"No matter how paranoid you are, what they're actually doing is worse than you can possibly imagine." - Ralph J. Gleason (1917-75)

Ideas?