

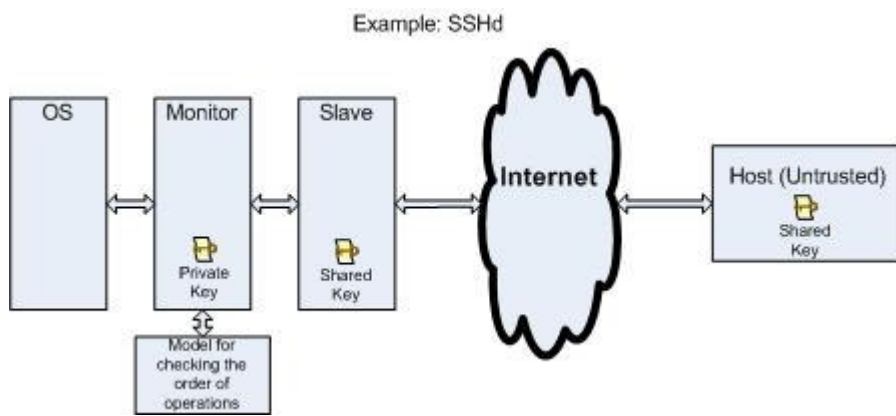
## Privilege separation

UNIX/Windows daemons handle many different operations. By the principle of least privilege, each operation should only be given the exact privileges that it needs.

Why do programs need privilege?

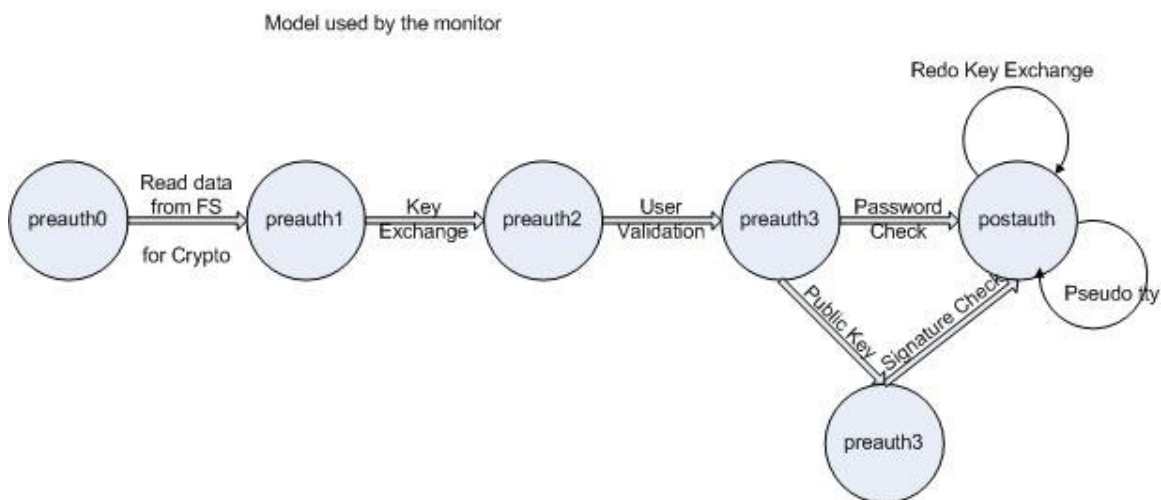
- To access secret data.
- To perform special operations
- To change identity (uid/gid)

From one program, we get a monitor and a slave. The monitor is small, trusted and privileged, the slave is larger untrusted and unprivileged. An attacker can gain control over the nobody-privileged slave program.



Even if the untrusted host is able to compromise the Slave process, she still only has access to the Shared Key (which she already had)

The original program will only perform privileged operations in the correct order. This order is given by a certain FSA. For OpenSSH:



Which is safer?

- Slave executes monitor
- Monitor executes slave.

The transformation from the monolithic program to the monitor/slave should not add/remove any bugs.

If the slave executes the monitor, there is little stopping another client from communicating directly with the monitor. If a bug lands in the monitor code, it is exploitable by a remote attacker. In this scheme, we have not protected the privileged operations at all.

If the monitor executes the slave, we may or may not be safe. There still may be a bug in the monitor, but it is hard to exploit unless there is also a bug in the slave. The slave provides the only interface to the monitor.

Automatic Privilege Separation Privtrans.

- Annotate Functions
- Annotate variables
- Type qualifier inference
- Storage subsystem in monitor
- RPC

The Privtrans paper was relatively vague about the implementation of the function/variable annotation and the type qualifier inference scheme used.