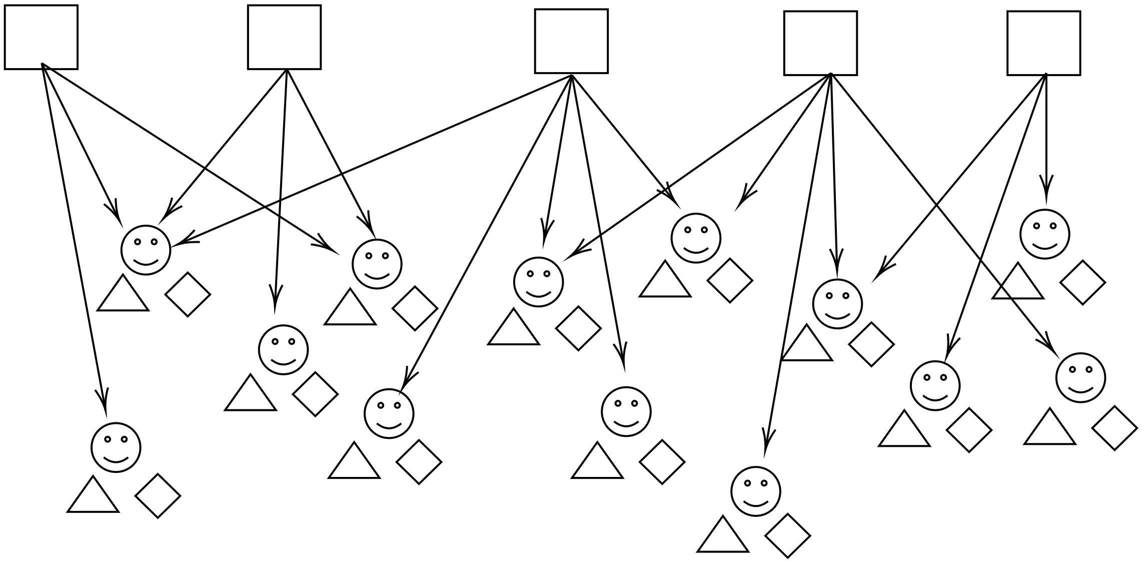
COSC 4765
Computer Security

Name: Jacob Tuttle

# Homework 07

**November 1, 2020**

**A depiction of the System**



Please note the system doesn't show every connection or patient for sake of clarity. In the system described, each machine would be monitoring each person, and there would be 25 people each associated with a triangle and diamond resource.

**Possible Attack**

One possible attack against this system could be a cinderella attack against one of the medicine dispenser resources. Since the systems are distributed and there isn't a single machine telling a dispenser when to dispense (a situation that would bring its own issues of course) an adversary could skip the time on the machine forward to the next dispensing time repeatedly. This could allow all of the medicine to be drained from a dispenser allowing the attackers to steal the excess.

### Replay Attacks

To prevent replay attacks, you could implement a small challenge/response action between the peripheral medical devices and the monitoring devices. Before being allowed to perform an action, the peripheral device would send something like an encrypted time stamp and the monitoring device could respond with an encrypted derived from that time stamp. As long as the encryption system between the two was secure, and both devices could maintain a properly list of their recieved and sent nonces, replay attacks could be prevented.

### Race Conditions

To prevent race conditions, all devices involved could have to wait for a ”token.” Say two peripheral medical devices each need to dispense a drug into a patient, but they can’t both dispense at the same time. Whichever of them last acted would have a token, made up of some shared secret between them. If another device connected to them asking for the token, they would finish any pending actions they could take and then hand off the token to the other device. This implementation could be vulnerable to a malicious actor simply taking the token and never relenquishing it; however, we could work to prevent this by validating the user attempting to take the token once again through some form of shared secret (i.e. we exchange encryption keys, having different ones for each device, and if we’re able to decrypt the request, we have already trusted the agent requesting the token).