Intelligent Agents of Intelligence fight for Intelligence

1.
    1. F128: read access
    2. F360: write access
    3. F522: read and write access
    4. F513: read access
    5. F119: no access
    6. F904: read access
    7. F100: no access
    8. F346: write access
2.
    a) Ray: Lowest: (Handler, {α, γ, δ}) Highest: (Director, {α, γ, δ, η, λ})
    b) Cyril: Lowest: (Handler, {α, β, δ, η}) Highest: (Handler, {α, β, γ, δ, η, λ})
    c) Cyril. Since Cyril's lowest clearance level and minimal set of compartments are already fulfill the requirement for reading F999.

3. The order of access:
    a) Read F129
    b) Read F676
    c) Read F513
    d) Write F123
    e) Read F369
    f) Write F123
    g) Read F101
    h) Write F123

Securing password authentication

1. No. User use the salt the wrong way(salt should be append to password before pass into the hash function). Attacker could exploit the server and get access to the file that store the fingerprint and salt for each user. Then attacker prepare a single table of fingerprints and password or just use the online decode websites. And for each fingerprint in the file obtained from the server back-end, update the fingerprint by subtract the corresponding salt. Now we have the original hash value. The attacker could search for matching fingerprint in the pre-built table or online. Attacker maybe able to exploit the original password in really short time.

2. Before pass the password into hash function, pad the 8-bit salt to the password. Or even add the salt in certain location of the password, add in a different order and etc.

3. The hash was generated using sha1 hash function. And the password is "petunia". I found a website that try all hash function and decoded this in less than 1 min.

4. If prefer SHA related hash function, sha256 is more secure than sha1. since it introduce more bits and it have way more possible combinations than sha1. Another hash function is bcrypt which is very expensive to calculate for both server and attacker which means less possibility of using rainbow table to decode.

Firing up the Firewall:

1. No. It is relatively easy for attacker to notice that the IP has been blocked and find a new IP and perform attack again. Should use white list which only allow certain access.

2. Spoofed/smurf Attack: attacker spoof address of sender end node in packet by setting it to victim's address(in this case ip within internal network)
Should use packet filtering gateways.

3. Rules:

a) Allow 32.23.11.0/25 => 0.0.0.0/0 From PORT 80 to all BY TCP
b) Allow 0.0.0.0/0 => 32.23.11.25 From PORT all to 443 BY TCP
c) Allow 0.0.0.0/0 => 32.23.11.0/25 From PORT all to 22 BY TCP
d) Allow 32.23.11.0/25 => 53.16.71.12 From PORT 5000 - 5100 to 1551 BY BOTH
e) Allow 9.19.11.217 => 32.23.11.10 From PORT all to 3223 BY TCP
f) Allow 32.23.11.0/25 => 32.23.11.10 From PORT all to 3223 BY TCP