```
(*
  Identificação do autor, data e versão do programa
*)

(* Carregar módulos necessários para a execução *)
#load "str.cma"

(* Passo 1 - Leitura dos ficheiros *)

(* Os ficheiros podem ser dados como parâmetro *)
(*
let f_passwd = Sys.argv.(1);;
let f_group = Sys.argv.(2);;
*)

(* Os ficheiros podem ser especificados em variáveis *)
let f_passwd = "files_aux/02_etc_passwd.txt" ;;
let f_group  = "files_aux/02_etc_group.txt" ;;

(*
  Description: Função recursiva para a abertura e leitura de um ficheiro.
  Input: name - Nome do ficheiro
  Output: List de strings com o conteúdo do ficheiro
*)
let read_lines name : string list =
  if Sys.file_exists (name) then
    begin
      let ic = open_in name in
      try
        let try_read () =
          try Some (input_line ic) with End_of_file -> None in
        let rec loop acc = match try_read () with
          | Some s -> loop (s :: acc)
          | None -> close_in_noerr ic; List.rev acc in
        loop []
      with e ->
        close_in_noerr ic;
        []
    end
   else
    []
  ;;


type user =
{
    name : string;
    password : string;
    uid : int;
    gid: int;
    comment: string;
    directory : string;
    shell : string;
  user_type : string;
};;


(*
Função para separar a informação de uma listas
Input: (opcional) separador, lista
Output: lista de argumentos separados
*)
let rec splitinfo ?(sep=":") l  = match l with
  | []  -> []
  | x::xs ->  (Str.split (Str.regexp ":") x)::splitinfo xs;;
```

```
(*
Função de verificação de shells de um utilizador, caso não seja esteja ativa retorna
Input: nome de uma shell
Output: 1 - Shell activa, 0 - Inativa (não tem shell)
*)
let check_has_shell shell:int =
  match shell with
    | "" -> 0
    | "/sbin/nologin" -> 0
    | "/usr/sbin/nologin" -> 0
    | "/sbin/false" -> 0
    | "/usr/sbin/false" -> 0
    | "/dev/null" -> 0
    | "/bin/false" -> 0
    | _ -> 1 ;;


(*
Função para ir buscar as posicoes do utilizador
Input: lista
Output: user numa estrutura tipo user
*)
let get_user l:user =
  let uid =  int_of_string (List.nth l 2) in
  let user_type = if  uid > 100 then "user" else "sys" in
  let user_name = List.nth l 0 in
  {
    name = user_name;
    password = List.nth l 1;
    uid = uid;
    gid = int_of_string (List.nth l 3);
    comment = List.nth l 4;
    directory = List.nth l 5;
    shell = List.nth l 6;
    user_type = user_type;
  };;


(*
Função para verificar a vulnerabilidade 2
Input: User
Output: 1 - é root e tem shell
        0 - não tem shell
*)
let vuln1 u:int =
  if u.name = "root" && check_has_shell u.shell = 1 then
    1
  else
    0;;

(*
Função para verificar a vulnerabilidade 2
Input: User
Output: 1 - tem o uid a 0 ou i gid a zero
        0 - não tem, é user com id normal
*)
let vuln2 u:int =
  if (u.uid = 0 || u.gid =0 ) && u.name <>"root" && u.user_type = "user" then
    1
  else
    0;;

(*
Função de reporting
*)
let report_statistics list_users list_groups =
  let risco_alto = ref 0 in
  let risco_baixo = ref 0 in
  let risco_medio = ref 0 in
```

```
    print_string (" ----- -- Analise de Vulnerabilidades -- ----- \n" ) ;
      List.iter  (
        fun user_l -> (
          match user_l with
          | [] -> print_string("> sem utilizador <\n")
          | _ ->
            let user = get_user user_l in
            if vuln1 user = 1 then (
              print_string (" "^ user.name ^" --> Vul. 1 \n" ) ;
              risco_alto := !risco_alto + 1;
            ) else
              if (vuln2 user = 1) then (
                print_string (" "^ user.name ^" --> Vul. 2 \n" ) ;
                risco_alto := !risco_alto + 1;
              )
              else
                print_string (" "^ user.name ^" --> OK \n"  );
            )
        ) list_users;
      print_string ("------- Resumo ---- ");
      print_newline();
      print_string ("Alto Risco " );
      print_int(!risco_alto);
      print_newline();
      print_string ("Medio Risco ") ;
      print_int(!risco_medio);
      print_newline();
      print_string ("Baixo Risco ");
      print_int(!risco_baixo);
      print_newline();
      ;;



(* ---------------------------------------------
Processmento principal
--------------------------------------------- *)
let main =
  let users_raw = read_lines f_passwd in
  let groups_raw = read_lines f_group in
  let list_users = splitinfo users_raw in
  let list_groups = splitinfo ~sep:":" groups_raw in
  ignore(Sys.command "clear");
  report_statistics list_users list_groups;;

```