E-mail sistêmico

Manuais relacionados à utilização e configuração de endereço de e-mail utilizados em aplicações.

- Recuperação/alteração senha e-mail sistêmico
- Configurações SMTP para e-mail de sistemas (não-responda.ifsp.edu.br)

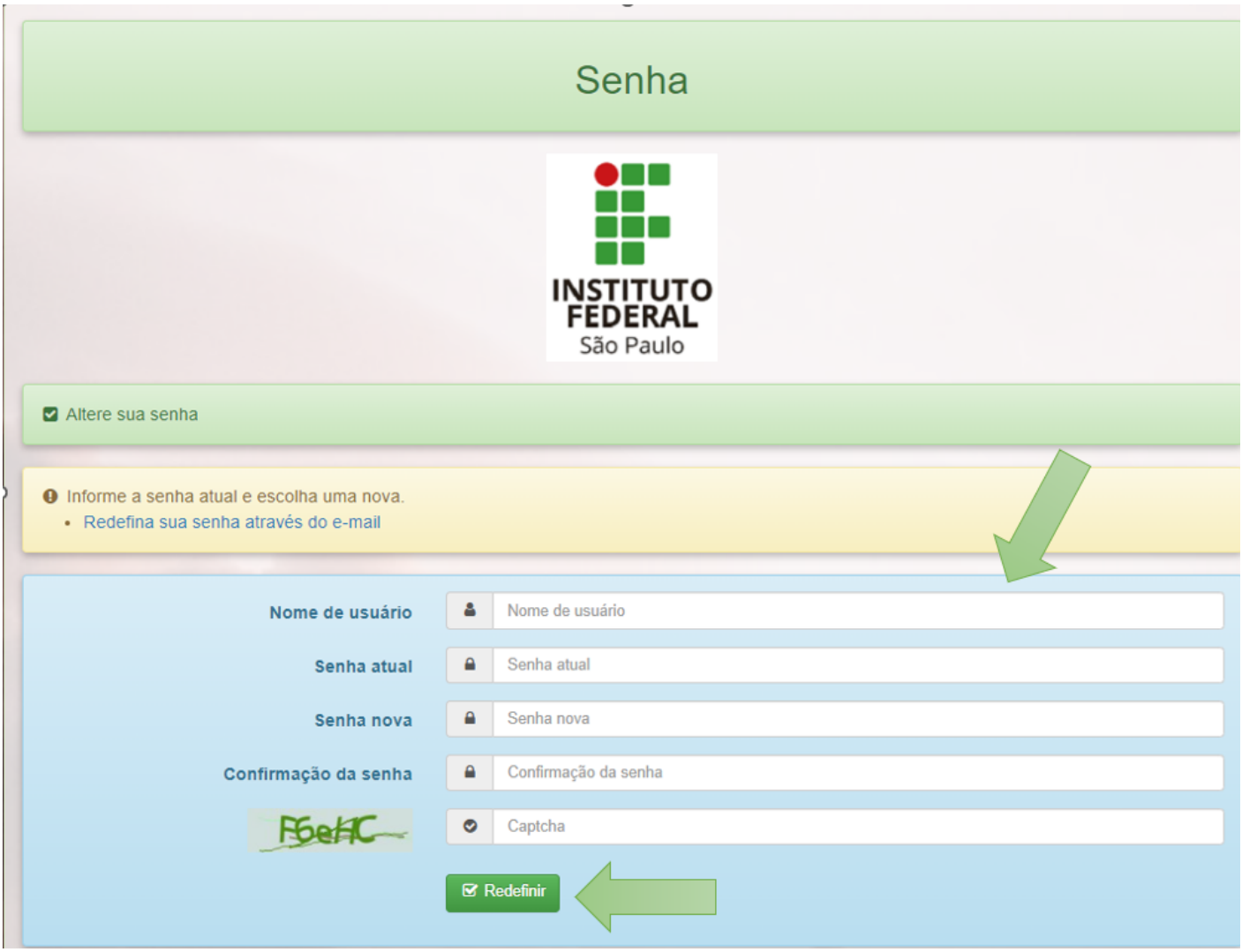
Recuperação/alteração senha e-mail sistêmico

> Tutorial para recuperação ou alteração da senha de e-mails utilizados em aplicações

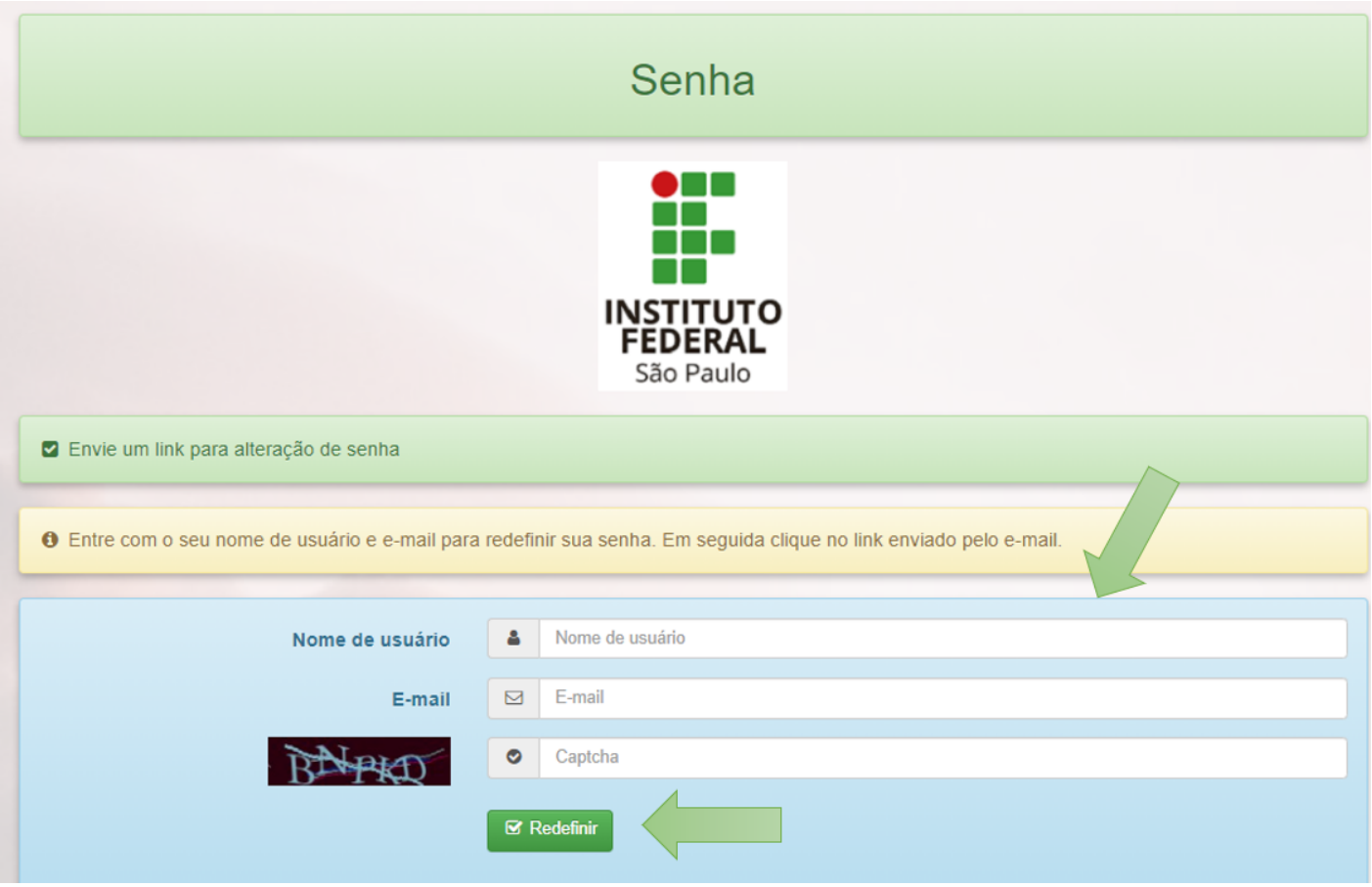
1. Após conectar à VPN acesse o link https://ssp.ifsp.edu.br/

> Importante: utilize a VPN do IFSP. A VPN interna do campus não irá permitir acesso ao endereço.

2. Caso saiba a senha, preencha os campos conforme informado e clique em `Redefinir`:



3. Se não possuir a senha anterior clique em Redefina sua senha através do e-mail. Em seguida preencha os campos:



O nome de usuário é o prontuário cadastrado para o e-mail sistêmico. Ex.: RT000236

O e-mail é o endereço institucional cadastrado para receber o link de alteração da senha.

Se necessário entre em contato com o suporte da reitoria para informar um novo endereço de e-mail para receber a notificação de recuperação.

Lembrando que o endereço não será criado/sincronizado com a Google até que esse procedimento de alteração seja executado pelo solicitante.

Configurações SMTP para e-mail de sistemas (não-responda.ifsp.edu.br)

Configurações de E-mail - Envio e recebimento de mensagens

Seguem abaixo as informações necessárias para configuração do e-mail institucional em serviços web (Ex.: GLPI, Moodle, etc).

Originalmente, o Simple Mail Transfer Protocol (SMTP) usava a porta 25. Atualmente, o SMTP deve usar a porta 587. Esta é a porta para transmissões de e-mails criptografados usando SMTP Secure (SMTPS).

Às vezes a porta 465 também é usada para SMTPS. Entretanto, esta é uma implementação ultrapassada e a porta 587 deve ser usada, se possível.

> Portanto: dar preferencia a **porta 587** com TLS; e usar a **porta 465** com SSL em casos que não for possivel usar a 587.
> A **porta 25** foi bloqueada por default e não é mais utilizada.

HOST: nao-responda.ifsp.edu.br
Autenticação: prontuario@nao-responda.ifsp.edu.br
Segurança: SSL/TLS
Portas: 465 ou 587 (preferivel).

Abaixo as portas geralmente listadas no servidor zimbra.

> “
> ```
> root@email:~# nmap  nao-responda.ifsp.edu.br
>
> Starting Nmap 5.21 ( http://nmap.org ) at 2020-10-28 12:30 AMST
> Nmap scan report for email.ifsp.edu.br (45.236.121.45)
> Host is up (0.000020s latency).
> Not shown: 986 closed ports
> PORT     STATE SERVICE
> 22/tcp   open  ssh
> 25/tcp   open  smtp
> ```

```
80/tcp   open  http
110/tcp  open  pop3
111/tcp  open  rpcbind
143/tcp  open  imap
389/tcp  open  ldap
443/tcp  open  https
465/tcp  open  smtps
587/tcp  open  submission
993/tcp  open  imaps
995/tcp  open  pop3s
5222/tcp open  unknown
7025/tcp open  unknown
```

Envio de E-mail via script

Casa seja necessário validar o envio de mensagens a apartir de uma determinada VLAN, pode ser usado o script abaixo:

> Pode ser necessário instalar o pacote dnspyhton.
> Ex.: python -m pip install dnspython

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import socket
import ssl
import smtplib
import dns.resolver
from datetime import datetime
import getpass
import sys

SMTP_SERVER = "nao-responda.ifsp.edu.br"
PORTS = [465, 587]
TIMEOUT = 10
LOG_FILE = "smtp_test.log"
```

```
def log(msg):
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    line = f"[{timestamp}] {msg}"
    print(line)
    with open(LOG_FILE, "a", encoding="utf-8") as f:
        f.write(line + "\n")


def test_dns():
    log("=== Teste DNS ===")
    try:
        answers = dns.resolver.resolve(SMTP_SERVER, "A")
        for rdata in answers:
            log(f"Registro A: {rdata.address}")
    except Exception as e:
        log(f"Erro DNS A: {e}")

    try:
        answers = dns.resolver.resolve(SMTP_SERVER, "MX")
        for rdata in answers:
            log(f"Registro MX: {rdata.exchange} (prio {rdata.preference})")
    except Exception as e:
        log(f"Erro DNS MX: {e}")


def test_tcp(port):
    log(f"=== Teste TCP porta {port} ===")
    try:
        with socket.create_connection((SMTP_SERVER, port), timeout=TIMEOUT):
            log(f"Porta {port} acessível")
    except Exception as e:
        log(f"Falha na porta {port}: {e}")


def test_banner_smtps():
    log("=== Teste Banner SMTPS (porta 465) ===")
    try:
        context = ssl.create_default_context()
        with socket.create_connection((SMTP_SERVER, 465), timeout=TIMEOUT) as sock:
            with context.wrap_socket(sock, server_hostname=SMTP_SERVER) as ssock:
                banner = ssock.recv(1024).decode(errors="ignore").strip()
```

```
                log(f"Banner recebido: {banner}")
    except Exception as e:
        log(f"Erro ao obter banner SMTPS: {e}")


def test_smtp_plain_587():
    log("=== Teste SMTP simples (EHLO) porta 587 ===")
    try:
        server = smtplib.SMTP(SMTP_SERVER, 587, timeout=TIMEOUT)
        server.ehlo()
        log("EHLO executado com sucesso")
        server.quit()
    except Exception as e:
        log(f"Erro SMTP simples (587): {e}")


def test_starttls():
    log("=== Teste SMTP STARTTLS (porta 587) ===")
    try:
        server = smtplib.SMTP(SMTP_SERVER, 587, timeout=TIMEOUT)
        server.ehlo()
        server.starttls(context=ssl.create_default_context())
        server.ehlo()
        log("STARTTLS negociado com sucesso")
        server.quit()
    except Exception as e:
        log(f"Erro STARTTLS: {e}")


def test_smtps():
    log("=== Teste SMTPS (porta 465) ===")
    try:
        context = ssl.create_default_context()
        server = smtplib.SMTP_SSL(SMTP_SERVER, 465, timeout=TIMEOUT, context=context)
        server.ehlo()
        log("Conexão SMTPS estabelecida com sucesso")
        server.quit()
    except Exception as e:
        log(f"Erro SMTPS: {e}")
```

```
def interactive_auth_test():
    answer = input("\nDeseja realizar o teste de envio SMTP autenticado? (s/N):
").strip().lower()
    if answer != "s":
        log("Teste de envio autenticado ignorado pelo usuário")
        return

    log("=== Teste de envio SMTP autenticado (interativo) ===")

    smtp_user = input("Usuário SMTP: ").strip()
    smtp_pass = getpass.getpass("Senha SMTP (não será exibida): ")

    mail_from = input("E-mail remetente: ").strip()
    mail_to = input("E-mail destinatário: ").strip()

    try:
        msg = f"""From: {mail_from}
To: {mail_to}
Subject: Teste SMTP Python (Interativo)

Mensagem de teste enviada via script Python SMTP interativo.
"""

        server = smtplib.SMTP(SMTP_SERVER, 587, timeout=TIMEOUT)
        server.ehlo()
        server.starttls(context=ssl.create_default_context())
        server.login(smtp_user, smtp_pass)
        server.sendmail(mail_from, [mail_to], msg)
        server.quit()

        log("Mensagem enviada com sucesso (SMTP autenticado)")
    except Exception as e:
        log(f"Erro no envio autenticado: {e}")


def main():
    log("===== INÍCIO DOS TESTES SMTP =====")

    test_dns()

    for port in PORTS:
```

```
        test_tcp(port)

    test_banner_smtps()
    test_smtp_plain_587()
    test_starttls()
    test_smtps()
    interactive_auth_test()

    log("===== FIM DOS TESTES SMTP =====")


if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        log("Execução interrompida pelo usuário")
        sys.exit(1)
```

Referência

- Cloudflare