```
Message-ID: <CAFRnB2Wyc9uLMz8O-YLQ3JZ1-fUYWr+NpFecYyFYdA1YyB+sfA@mail.gmail.com>
Date: Wed, 2 Nov 2022 08:03:31 -0400
From: Alex Gaynor <alex.gaynor@...il.com>
To: oss-security@...ts.openwall.com
Subject: Re: Re: OpenSSL X.509 Email Address 4-byte Buffer
 Overflow (CVE-2022-3602), X.509 Email Address Variable Length Buffer Overflow (CVE-2022-3786)

In Rust, assuming you wrote normal safe Rust[0], and you had code that
overran a buffer on the stack, you'd get a panic() -- which is roughly
an abort (there's even a mode where it literally is an abort. By
default it unwinds and runs destructors and such). As a general rule,
bounds check issues aren't caught at compile time (in contrast with
temporal safety, which mostly is enforced at compile time.)

Alex

[0]: Rust also has an `unsafe` keyword that lets you do unchecked
things with raw pointers. Using that for basic string manipulation
would be way outside of idiomatic Rust and I'd certainly expect it to
be flagged in code review.

On Wed, Nov 2, 2022 at 7:57 AM Tavis Ormandy <taviso@...il.com> wrote:
>
> On 2022-11-01, Jeffrey Walton wrote:
> > On Tue, Nov 1, 2022 at 3:55 PM Pavan Maddamsetti
> ><pavan.maddamsetti@...il.com> wrote:
> >>
> >> https://github.com/RustCrypto
> >
>
> I don't know rust, so serious question - if this same buggy punycode
> routine had been written in rust, what would have happened?
>
> - I assume you *could* write similar logic, but perhaps the argument is
>   that idiomatic rust discourages it?
> - Would rustc have been able to reason about the code well enough at
>   compile time to error out?
> - Just detect it at runtime and abort()?
>
> If the answer is "error out", then I think that's a pretty convincing win.
>
> Tavis.
>
> --
>  _o)            $ lynx lock.cmpxchg8b.com
>  /\\  _o)  _o)  $ finger taviso@....org
> _\_V _( ) _( )  @taviso
>


--
All that is necessary for evil to succeed is for good people to do nothing.
```