并行部署的改进型 SSL 工控 VPN 模型①

蔡照鹏 关 昕 (辽宁工程技术大学 电子与信息工程学院 辽宁 葫芦岛 125105)

摘 要: 工控网络中的通信实时性非常重要，但普通 SSLVPN 一直不能很好的满足其实时性需求，为此这里提出了一套改进 SSL 协议的措施，并依据这些措施构建了一个并行部署 2 台 SSLVPN 的新模型，分析了改进 SSL 协议的各个措施。通过引用基础数据计算出了模型改进前后的系统耗时，由此得出了新模型的实时性改善度为 1.5，时间节省率为 33%；最后补充介绍了该模型有待进一步研究的一些技术课题，展望了其应用前景。

关键词: 并行部署；改进型 SSL；工控 VPN；模型研究

An Improved SSL Industrial Control VPN Model under the Parallel Deployment

CAI Zhao-Peng, GUAN Xin

(School of Electronic and Information Engineering, Liaoning Technology University, Huludao 125105, China)

Abstract: Real-time communication is very important in the industrial control network, but a general SSLVPN is sometimes unable to meet this requirement perfectly. So a set of strategies are introduced to improve SSL protocol, on which a new parallel deploying 2-SSLVPN model is buit. This paper analyzes strategies and calculates the old and new models' time-consumption with the reference data. It has proved that the real-time improvement rate of the new model is 1.5 and its time-saving rate is 33%. It also introduces some future technical problems and their applications.

Keywords: parallel deployment; improved SSL; industrial control VPN; model research

随着 Internet 的快速发展，其高效统一的优良特性使之具有了进入工控领域的技术优势，同时，现代化大型工控领域也急需引入更高实时性、更易维护的跨大区域集中管理的先进技术，正如所需，普通网络技术中的 SSLVPN 就成了我们的首选；然而，对于一般企业来讲直接部署一台高价 SSLVPN，其性价比就尤显偏低了。

为降低企业的部署成本，提升低价位 SSLVPN 的实时性，这里我们提出了一种 SSLVPN 新模型：通过为标准 SSL 协议添加“忙”标志、引入优化算法(包括：KoblitzECC[1]、LZO[2]、SHA-1[3]、RC5[4]等)、附加 AddRAM 以及增加警告规范等措施形成一个改进型 SSL，再依据此 SSL 并配合其他策略对两台或多台低价 SSLVPN 作并行部署而构成的一个高实时性工控 VPN 新模型。下面我们就来详细的介绍一下该模型。

1 新模型分析

1.1 协议与模型的新旧对比

首先是，改进前的标准 SSL 协议结构[5]如表 1 所示。

表 1 标准 SSL 协议结构

…	握手字段	记录字段	警告字段	…

再者是，改进型 SSL 协议结构如表 2 所示。

表 2 改进型 SSL 协议结构

…	优化算法下的握手字段 “忙”标志	优化算法下的记录字段	增加新警告规范的警告字段	…
	握手层和记录层共用的 AddRAM			

① 基金项目:辽宁工程技术大学科研创新项目(Y200900505)
收稿时间:2009-05-16

根据上述两种协议结构分别可得以下两种部署模型：

标准 SSL 协议下的部署模型如图 1 所示，RemoteBrowser 直接连接单台 SSLVPN，然后 SSLVPN 再和总部的监控 Server 直接相连。

图 1　SSLVPN 的传统部署模型

而对 SSL 协议改进后，得到的并行部署 2 台 SSLVPN 的新模型如图 2 所示，RemoteBrowser 首先连接 VPNSelectServer(以下简称 VSS)上，VSS 经直接转发型 Hub 连接两台 SSLVPN，这两台 SSLVPN 经指令转接器再与总部监控 Server 相连。

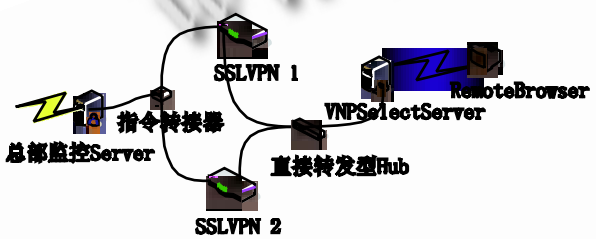

图 2　并行部署 2 台 SSLVPN 的新模型

新模型正是从硬件设备和软件措施两个方面来实现的；其中，硬件设备主要包括：如图 2 所示的 VSS、直接转发型 Hub 和指令转接器；而软件措施主要是指对 SSL 协议的改进及其辅助程序(即下文的设备程序)等。

对于硬件设备，现在市面上早已有价格合理的成型产品，其原理我们容易理解；而对软件措施的把握重点就是对改进 SSL 协议各个措施的理解了，下面我们就来重点分析一下这些措施。

1.2 协议改进措施的分析

1.2.1 “忙”标志及其影响

这里的“忙”标志我们用“busy”变量来实现；而对该标志的获取，我们考虑让 VSS 主动以高频向 SSLVPN 读取，而不是让 SSLVPN 向 VSS 发送，其目的就是尽量不让 SSLVPN 做额外任务，而只做其核心业务。

当握手层有了该标志后，整体 SSLVPN 系统的工作机制就变成了：当 RemoteBrowser 接入单台 SSLVPN 的会话到了定额时，SSLVPN 的“忙”状态进程会自动将“忙”标志置“忙”；同时，外围的 VSS 程序始终以高频主动读取比如 VPN1 的“忙”标志，如果它发现 VPN1“忙”就自动将 RemoteBrowser 请求转至 VPN2，之后，所有监控请求均由指令转接器发送至总部监控 Server。

至于 SSLVPN 与 VSS 之间协议互异的理解问题，我们依据 OPC 标准通过自定义的 CMSComm()类来实现，而该类又借助了 MS C++的 MSComm 控件；此外，为解决各 SSLVPN 的负载均衡问题，我们实现了 VSS 类；这两个类(以下简称设备程序)的类 C++核心代码如下：

```
Class CMSComm()
{MSComm vpn_x_Com,vss_Com;
Comm_Control(vpn_x_Com,vss_Com)
{vpn_x_Com.SetCommPort(atoi(x));
vss_Com.SetCommPort(atoi(x)); …
vpn_x_Com.SetPortOpen();
vss_Com.SetPortOpen(); …
vpn_x_Com.InitialLabel(busy);
vss_Com.ReadInput(vpn_x_Com); …
}…
}
Class VSS()
{Bool Compare(vpn_1_weight,vpn_2_weight)
{…
}…
if Compare(vpn_1_weight,vpn_2_weight)
    return vpn_1;
else
    return vpn_2;
    }
```

参数说明：vpn_x_Com 代表某 SSLVPN 通信对象，vss_Com 代表 VSS 通信对象；atoi(x)表示设备程序端口；vpn_x 代表某 SSLVPN；vpn_x_weight 表示某 SSLVPN 负载。

另外，还需要说明的是，新模型中 SSLVPN 的部署台数是可调的，如需多台我们可对设备程序仅作简单修改，就可满足多接口的互理解和多任务的智能调度等需求。

1.2.2 优化算法和 AddRAM 的作用

由于在握手层我们采用了运算速度更快的 KoblitzECC 算法，因此新模型在建立握手时，就能以更快的速度完成身份认证、密钥交换等工作。

当新模型引入 AddRAM 后，SSLVPN 就形成了三级存储体系：内部 R、内置 RAM 和附加 AddRAM；其目的主要有三个：其一，为 SSL 各层协议程序提供逐一缓存空间，大幅度提升运行效率；其二，为 SSL 握手的 Keeping Connection 字段提供存储空间，提升握手复用率；其三，为 SSLVPNCPU 中断请求队列提供存储空间，实现对外部请求的动态快速响应。

更重要的是，此时的记录层就能与握手层共用 AddRAM，这样，记录层的工作过程就变成了：首先把接收的信息分割成可管理的数据块；然后把这些数据块依次装入 AddRAM、内置 RAM、内部 R；之后采用目前压缩效率最高的 LZO 算法对其进行压缩；接着用能产生最优长度的 SHA-1 算法产生 MAC，用以检验数据的完整性；再用目前加密效率最快的 RC5 算法对其加密；最后把结果送至应用层。

因此，从握手层和记录层这样的工作环境来看，我们引入的优化算法和 AddRAM 确实能显著提升系统的整体实时性。

1.2.3 新警告规范及其必要性

尽管新模型的核心目标是实现高实时性，但从用户角度看，除了高实时性外，当系统出现某些故障时，能否将工控现场的状况以“最优”完整性数据表示出来，并以较形象、高精确的手段实时呈现给管理层，就显得尤为重要了。

所以，我们有必要为原警告协议的组态标准添加一些新规范，比如：工控现场的警告紧急级别、确认标志、当前值和限值的制约关系等项目，以及关于这些项目的动态界定规则，并重点增强对 HMI 高级语言的支持能力。

其中，当前值和限值的制约关系，我们考虑采用模糊隶属变量来表示当前值距离限值的隶属距离，这样就可根据工控设备的实际组态来实时调整设备的警告容忍度，使其更符合工控现场的实际状况；而上述各项目的动态界定规则也是依据此变量实现的；此外，增强对 HMI 高级语言的支持能力，目的是让新警告规范具有扩展高实时性呈现机制的扩展接口，以最大限度的实现“所见即现场”的工控目标。

通过这些来进一步优化系统的实时性，以便为其提供友好的管理界面、便捷的操作手段以及待扩展的功能接口等。

2 新模型实时性改善度的验证

2.1 基础数据

本文以如表 3 所示的数据为参考数据。

表 3 某单台 SSLVPN 的主要性能指标

项目	性能指标(个)
SSLVPN 并发会话数	2000
每秒接收会话数	200
⋮	⋮

(数据来源：深圳市深信服电子科技有限公司[2])

假定远程用户在某时段内的并发会话任务数为 6000 个。

再设原单台 SSLVPN 接收最大 2000 个会话时的耗时为 T_0；这样，在单台情况下，就会有 4000 个会话处于等待，此时接收 6000 个会话的总耗时 $T_{old}=3T_0$。

而在新模型中，当一台 SSLVPN 达到最大 2000 个会话时，VSS 就会把另外的 2000 个会话转给另一台 SSL VPN，又因为我们引入了优化算法和 AddRAM，这样，两台 SSLVPN 在单纯接收 4000 个会话的耗时 T_{4000} 明显是小于 T_0 的(这里的 T_{4000} 不含下文的 T_{ov})；而剩余的 2000 个会话，VSS 会通过智能判断将之交给一个任务较少者比如 VPN1，同时当 VSS 发现 VPN2 任务变的较少时，又会自动将刚分给 VPN1 的任务重新再分配给任务较少的 VPN2，通过这样的智能调度使整个系统的任务配额总处于最优。

设新模型中的两台 SSLVPN 单纯接收 6000 个会话的耗时为 T_{00}，再假定设备程序的运行耗时为 T_{ov}，则新模型整体接收 6000 个会话的总耗时 $T_{new}=T_{00}+T_{ov}$。

2.2 重要结论：$T_{00}<<2T_0$

正如上文，新模型的两台 SSLVPN 接收 4000 个会话的耗时 T_{4000} 是小于 T_0 的。

而它们在共同接收剩余的 2000 个会话时，由于采用了智能调度，使两台 SSLVPN 以协作的方式接收了剩余任务；再者，即使我们仅让一台 SSLVPN 单独接收剩余任务，其耗时也肯定是小于 T_0 的(道理同上述的 T_{4000})；由此可知，对于接收剩余 2000 个会话的耗时肯定是远远小于 T_0 的。

因此可知：$T_{00}<<2T_0$。

但是，我们为了得到一个可信的验证结论，下文中我们不是直接引用上述结论的，而是将 T_{00} 最大化为 $2T_0$ 来验证的。

2.3 耗时对比

2.3.1 改进前的 T_{old}

原单台 SSLVPN 接收一个会话的耗时 T’为：

T’ =1s/200 个=5ms/个

其接收 2000 个会话的耗时 T_0 为：

T_0=2000T’ =10000ms

所以其接收 6000 个会话的耗时 T_{old} 为：

$T_{old}=3T_0=30000ms$

2.3.2 改进后的 T_{new}

由于 $T_{new}=T_{00}+T_{ov}$，我们首先来计算 T_{ov} ：

这里我们用测时程序 CGetPeriodTime()[6]的改进程序 GetRunTime()来测试 T_{ov}；其测试环境为：VSS 选用 Intel(R)Xeon(TM)CPU 3GHz 2.99GHz, DDRⅡ 4GB；编译环境为 MS VC++ 6.0；此外，为在模型环境下得到准确的 T_{ov}，我们依据深信服最新发布的 SINFORM5100-S 型 SSLVPN 的主要性能数据[2]给设备程序各参数赋以初始值。

GetRunTime()的类 C++主要代码如下：

```
Class GetRunTime
{Begin()
{QueryPerformanceCounter(&_PeriodTime._li); …
}
GetRunTime:: End()
{QueryPerformanceCounter(&li._li); …
return(_PeriodTime._time*NUMber)/s_InitFreQue
uency.s_Frequency);
  }
}
main()
{GetRunTime time;
begintime=time.Begin();
… //调用被测设备程序
cout<<time.End()-begintime<<endl;
}
```

经过这样的准备之后，下面就来实际计算 T_{ov} ：

运行上述测时程序 10 次，则设备程序各次运行的耗时如表 4 所示：

表 4　设备程序各次运行耗时表

序号(次)	耗时(ms)
1	0.32010
2	0.28901
3	0.29080
4	0.33010
5	0.31093
6	0.32997
7	0.32101
8	0.30079
9	0.28891
10	0.32048

则运行 10 次的耗时总和 T_{10}=3.10210ms

取其平均值，得：

$T_{ov}=T_{10}/10=0.31021ms$

其次，再把 T_{00} 取为最大：$T_{00}=2T_0$

最终得：

$T_{new}=T_{00}+T_{ov}=2T_0+T_{ov}=20000.31021ms$

2.4 结论

由上述计算结果，可知：

$T_{new}=20000.31021ms<<T_{old}=30000ms$

由此得，新模型的实时性改善度 $S=T_{old}/T_{new}\approx1.5$

其时间节省率 $P=(T_{old}-T_{new})/T_{old}\approx33\%$

这样，当我们不是将 T_{00} 假定为最大 $2T_0$，而是将 T_{00} 取为本模型的实际数值时(即取比 $2T_0$ 小很多的数值)，上述的 S 和 P 就会大幅度增大。

由此可知，新模型中仅仅并行部署两台低价 SSLVPN 的性价比，相对于部署同性能的单台高价 SSLVPN 来讲，其实时性改善度就远远超过了 1.5，时间节省率也明显大于 33%，那么在实际应用中，如果我们并行部署多台这样的 SSLVPN 所得的 S 和 P 显然就更加可观了。

因此，新模型很好的解决了原单台低价 SSLVPN 无法达到高实时性的问题；同时，通过增加警告规范，使得工控 SSLVPN 的前台监控模块更加智能更易扩展了。

3　结语

为满足工控企业对 SSLVPN 的高实时性需求，同时也兼顾其部署成本，本文提出了一个并行部署的改进型 SSL 工控 VPN 模型，通过验证证明了其实时性改善度是非常明显的，达到了预期的目标。当然，该模型在某些技术细节上，还可能存在这样或那样的问题，比如：模型主辅设备的速度瓶颈、多 SSLVPN 系统下身份认证和密钥管理的复杂性等问题仍需我们进一步

(下转第 96 页)

(上接第 100 页)

研究和解决。

但是，本文的一个重要的目的就是，希望通过我们提出的这么一个基本模型来引出相关的技术课题，以便相关领域的研究者和我们一道开展合作、共同探讨，把 SSLVPN 技术更好的应用到大型工控领域，以形成实时高效、成本合理、安全智能、界面友好的现代化工控网络系统，最终为我们的大型工业生产提供更加高效便捷的现代化管理手段。

参考文献

1 Koblitz N. Elliptic curve cryptostems. Mathematic of Computation, 1987,48(5):203 – 209.

2 深圳市深信服电子科技有限公司.Sinfor SSLVPN IPSEC/SSLVPN 一体化的选择.Vol.2.66:深圳:深圳市深信服电子科技有限公司, 2007.23 – 56.

3 Mandy Andress.计算机安全原理,北京:机械工业出版社, 2002.152 – 155.

4 ADAMSC, LLOYDS. Understanding Public key nfra structure: Concepts, Standars, and Deployment Consideratiods.Version One. Indiana: Macmillan Technical Publishing, 1999.33 – 37.

5 Rutgers M. Introduction to the Internet Protocols. Version One. New Jersey: The State University of New Jersey, 1997.48 – 49.

6 UcanCodeSoftware.The Research of the Measurement of a Program Running. 2006. http://www. ucancode. net / Free_C++_Source_Code-library.htm.