openssl ECDHE-RSA... cipher suite selected while using RSA server cert and ECDSA client cert

up vote 4 down vote favorite	The scene: s_server using RSA certificate. s_client using ECC-ECDSA certificate. Client authentication is required (option -Verify set on s_server) Cipher-suite selected after handshake is ECDHE-RSA-AES256-GCM-SHA384. So, i don't completely understand how the type of certificate (RSA-DSA) affects the cipher-suite selected (regarding the authentication part of the cipher-suite). Could it be that the server's certificate set the type of authentication used? Could someone explain how the server/client certificates affect the cipher-suite selection? certificates rsa cipher-selection ecc dsa
	add a comment

shareimprove this question	edited May 18 '15 at 7:53 Jens Erat 14.1k7 4163	asked Sep 24 '14 at 9:27 jpradas 2313

	I believe this answer relates to your question: security.stackexchange.com/q/65622/52676 – RoraZ Sep 24 '14 at 11:19

1 Answer

active oldest votes

up vote 6 down vote accepted	"ECDHE" means that the key exchange will use the Diffie-Hellman algorithm (over elliptic curves) with freshly generated DH elements; the last "E" stands for ephemeral. So while DH produces a shared key, it will work with randomly produced values, and nothing in DH will ensure authentication: the client has no way to know whether the DH public key it sees really belongs to the intended server. To bring back server authentication, something else is needed. It is a signature: the server digitally signs its half of the DH key exchange. That signature uses the key which is in the server's certificate, and thus the signature algorithm depends on the type of that key. In "ECDHE-RSA", the "RSA" part relates to that signature: it says that it will be of type RSA (and implies, of course, that the server's certificate contains the corresponding RSA public key). There is no such mechanism for client authentication. When the server requests a client certificate, the client will have to compute a signature and send its public key (as part of its certificate), but there is no relation between that client certificate type and the cipher suite. The reasons for that are: Client certificates appear later in the handshake, after the cipher suite selection. When the cipher suite is chosen, the client does not yet know whether a client certificate will be requested at all. A SSL server, in general, must be able to answer to the whole World, but when the client connects, it knows what server it is talking to, and can select an appropriate certificate in consequence (the server also sends a list of CA names to narrow down the client choices). Client certificates are a rarity in a Web context; in all generality, SSL servers ask for client certificates when they know that the clients already have certificates of an appropriate type (e.g. a bank server will ask a customer's certificate only if the customer is part of a program in which the bank ensured distribution of the said certificates). Therefore, there is no need for a dynamic negotiation of the client certificate type. The cipher suite selection is all about choosing algorithms which are supported at both ends. If the server must sign, and the client verify, then the client must be able to understand the signature algorithm and thus its key type. In practice, the client sends a list of supported cipher suites, and the server selects one of them -- a cipher suite that, of course, corresponds to the algorithms and key types that the server is going to use. Theoretically, the list sent by the client is ordered by "preference" and the server is supposed to honour that order, i.e. choose the first cipher suite in the client list that also fits the server's abilities. Not all servers are that courteous. See this answer for an introduction to SSL/TLS.

shareimprove this answer	edited Mar 17 at 10:46 Community♦ 1	answered Sep 24 '14 at 10:53 Thomas Pornin 250k45586813

	"the first "E" stands for ephemeral" It's actually the last E that stands for ephemeral, like in DHE – David 天宇 Wong May 4 '15 at 16:39
	Oh, indeed. Now fixed. – Thomas Pornin May 4 '15 at 18:44
	I corrected Thomas Pornin, I never thought this day would come – David 天宇 Wong May 4 '15 at 21:07

1	
